


# Unidade VII: Árvore Binária - Remoção



**PUC Minas**

Adaptação dos slides elaborados pelo Instituto de Ciências Exatas e  
Informática - Departamento de Ciência da Computação

- Funcionamento básico
- Algoritmo
- Análise de complexidade

- **Funcionamento básico** 
- Algoritmo
- Análise de complexidade

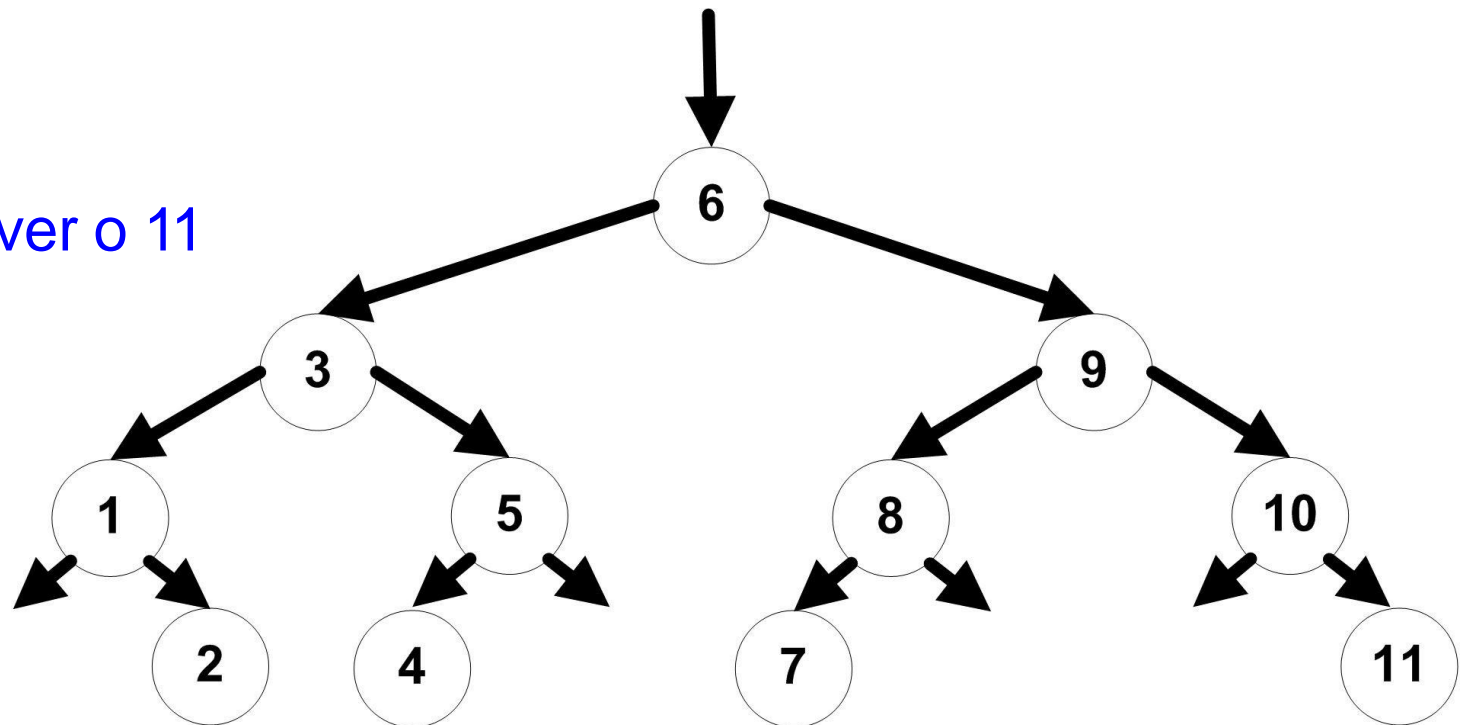
# Funcionamento Básico da Remoção

(1) Se o elemento estiver em uma **folha**, removê-la

# Funcionamento Básico da Remoção

(1) Se o elemento estiver em uma **folha**, removê-la

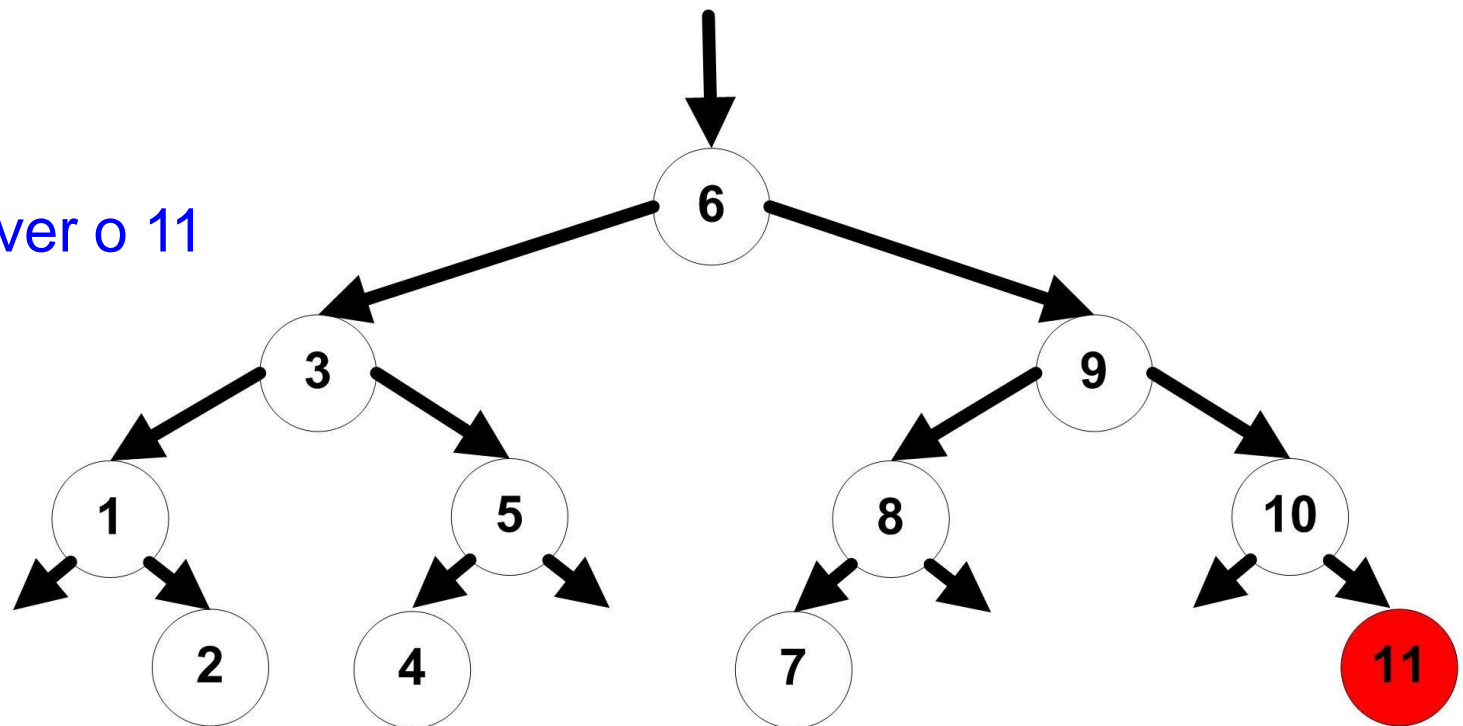
Exemplo: Remover o 11



# Funcionamento Básico da Remoção

(1) Se o elemento estiver em uma **folha**, removê-la

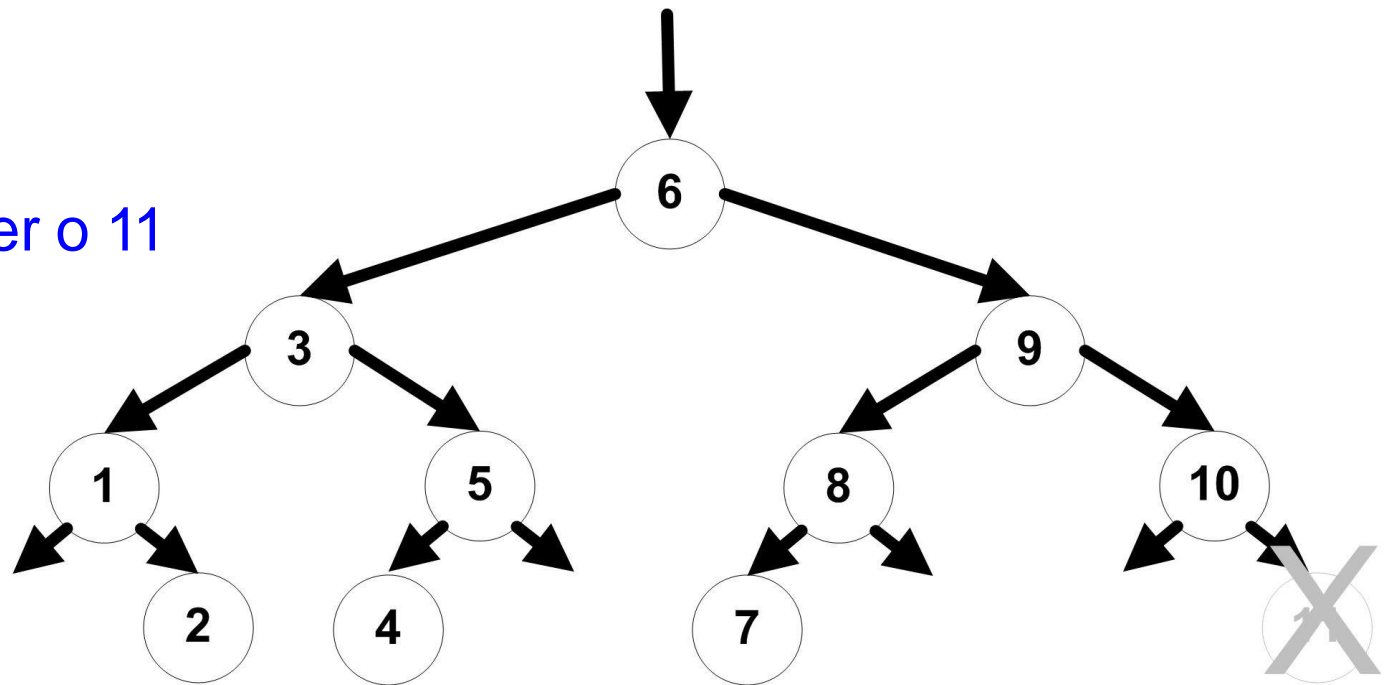
Exemplo: Remover o 11



# Funcionamento Básico da Remoção

(1) Se o elemento estiver em uma **folha**, removê-la

Exemplo: Remover o 11



# Funcionamento Básico da Remoção

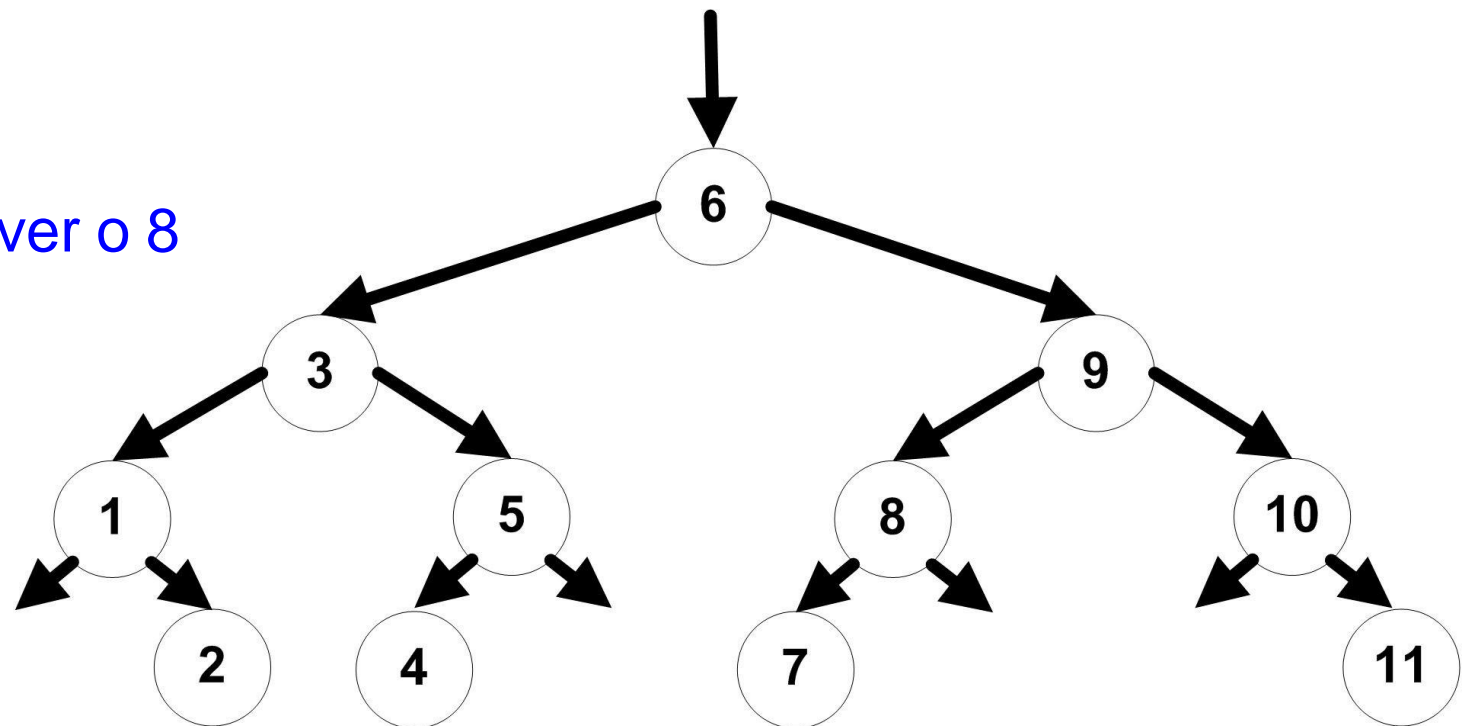
(2) Senão, se o elemento estiver em um **nó interno com um único filho**, Remover o nó e fazer com que seu pai aponte para seu filho



# Funcionamento Básico da Remoção

(2) Senão, se o elemento estiver em um **nó interno com um único filho**, Remover o nó e fazer com que seu pai aponte para seu filho

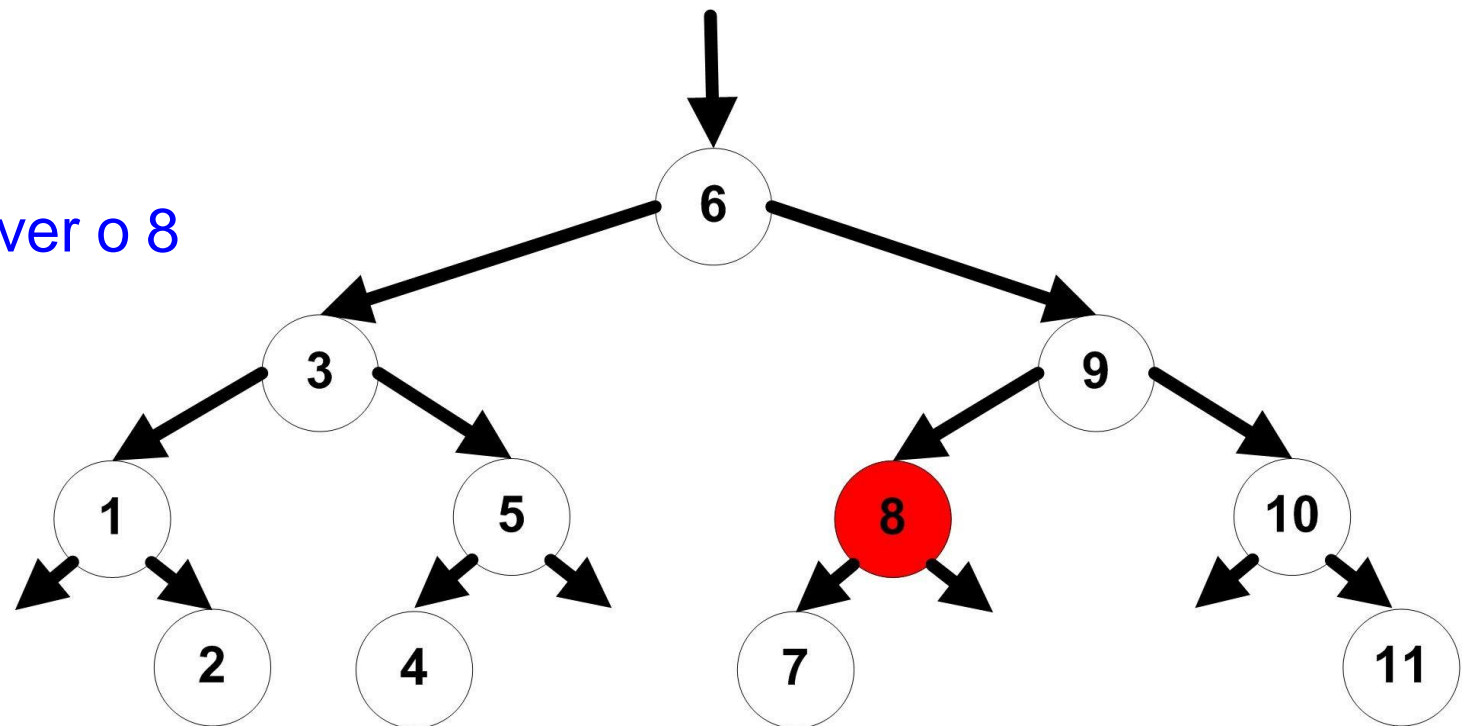
Exemplo: Remover o 8



# Funcionamento Básico da Remoção

(2) Senão, se o elemento estiver em um **nó interno com um único filho**, Remover o nó e fazer com que seu pai aponte para seu filho

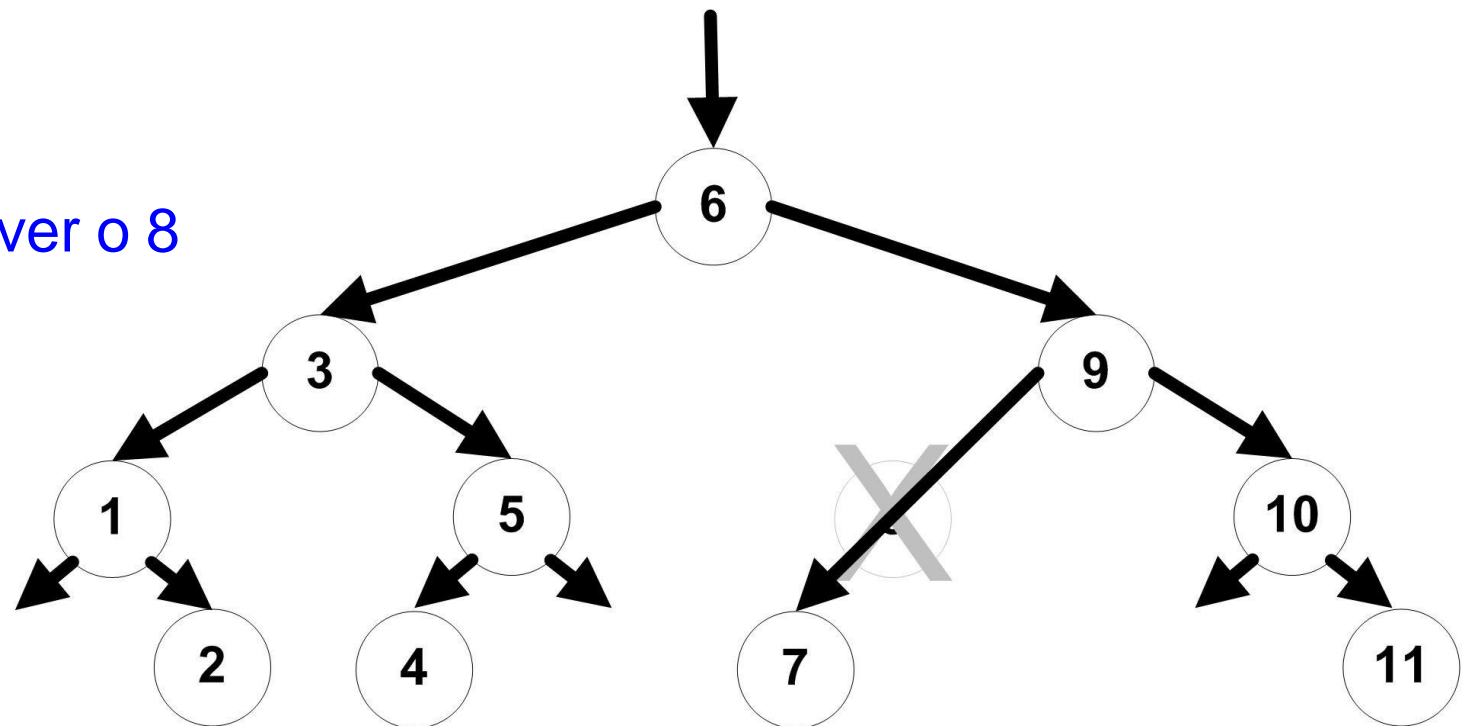
Exemplo: Remover o 8



# Funcionamento Básico da Remoção

(2) Senão, se o elemento estiver em um **nó interno com um único filho**, Remover o nó e fazer com que seu pai aponte para seu filho

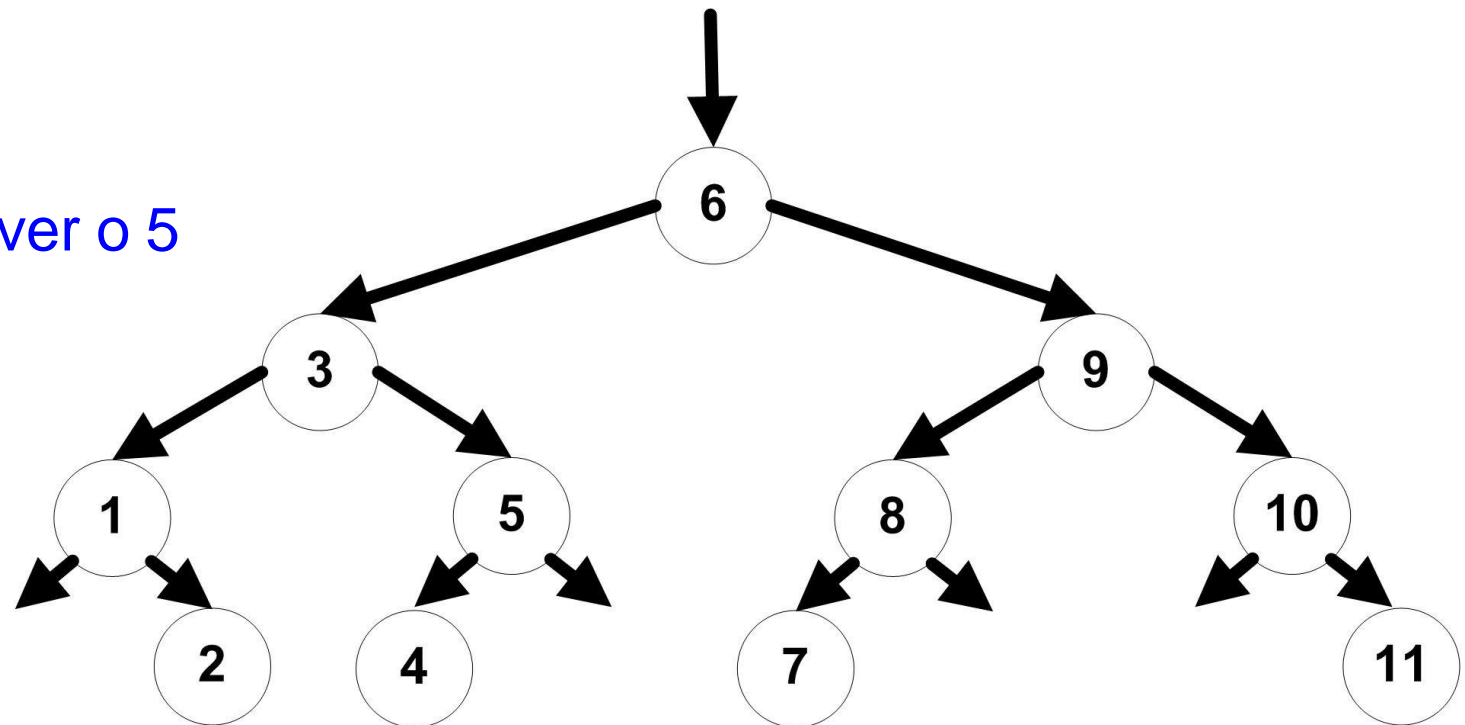
Exemplo: Remover o 8



# Funcionamento Básico da Remoção

(2) Senão, se o elemento estiver em um **nó interno com um único filho**, Remover o nó e fazer com que seu pai aponte para seu filho

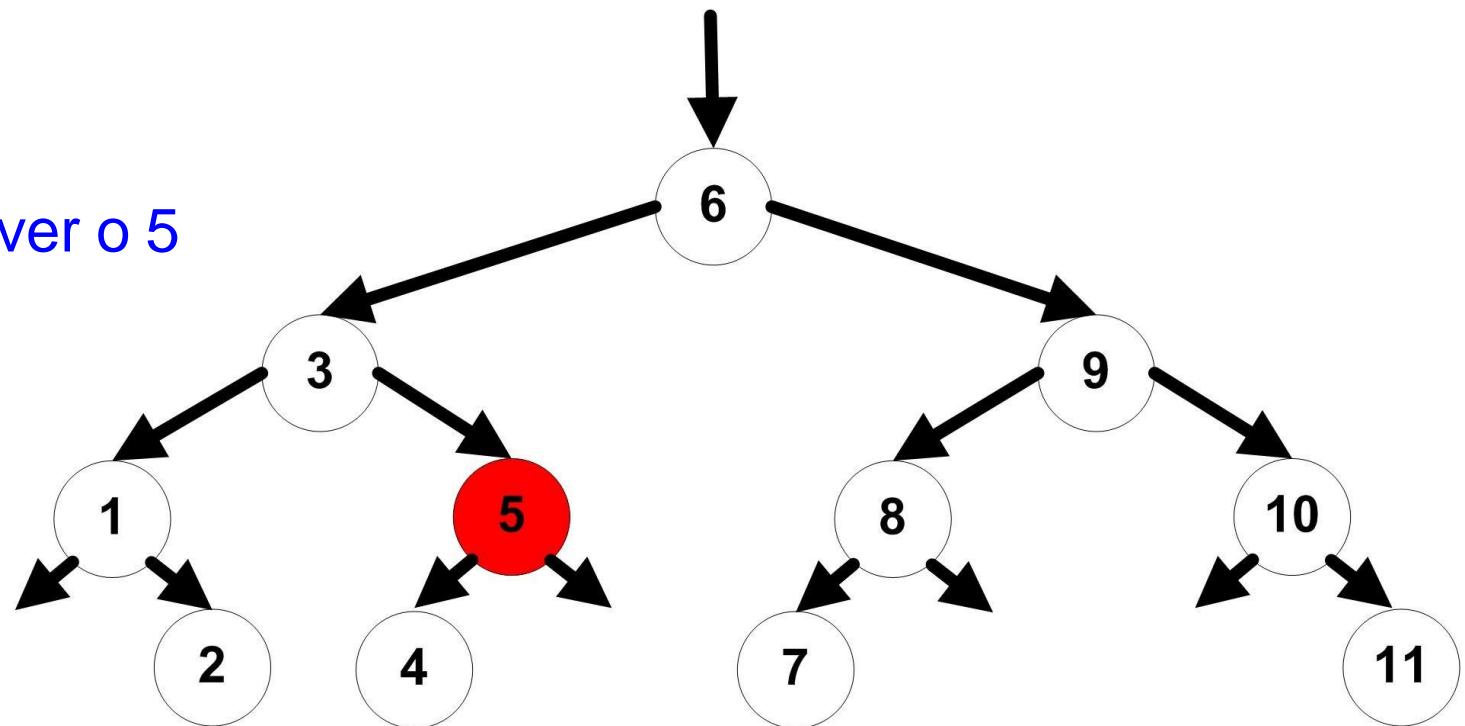
Exemplo: Remover o 5



# Funcionamento Básico da Remoção

(2) Senão, se o elemento estiver em um **nó interno com um único filho**, Remover o nó e fazer com que seu pai aponte para seu filho

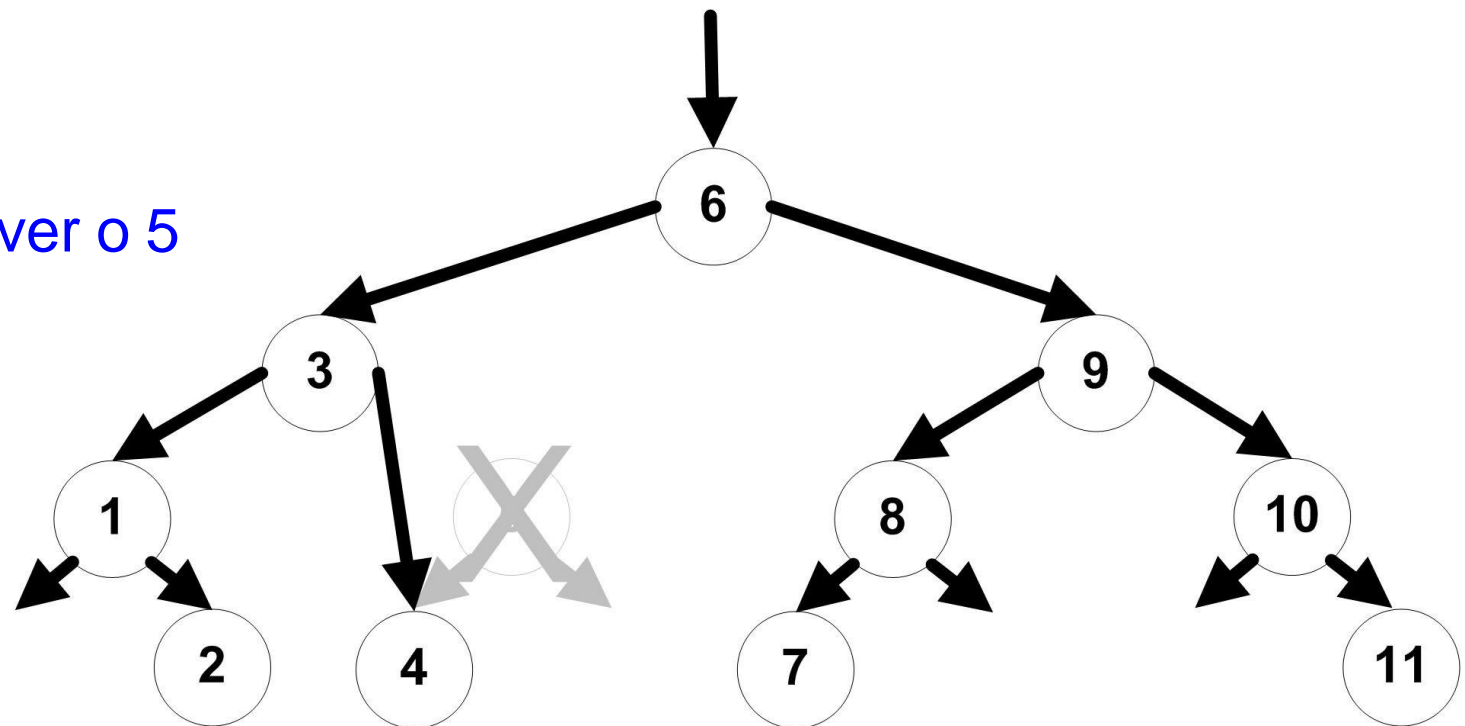
Exemplo: Remover o 5



# Funcionamento Básico da Remoção

(2) Senão, se o elemento estiver em um **nó intermediário com um único filho**, Remover o nó e fazer com que seu pai aponte para seu filho

Exemplo: Remover o 5



# Funcionamento Básico da Remoção

(3) Senão, se o elemento estiver em um **nó intermediário com dois filhos**, o elemento a ser removido deve ser substituído ou pelo **maior nó da subárvore à esquerda** ou **menor nó da subárvore à direita**

# Funcionamento Básico da Remoção

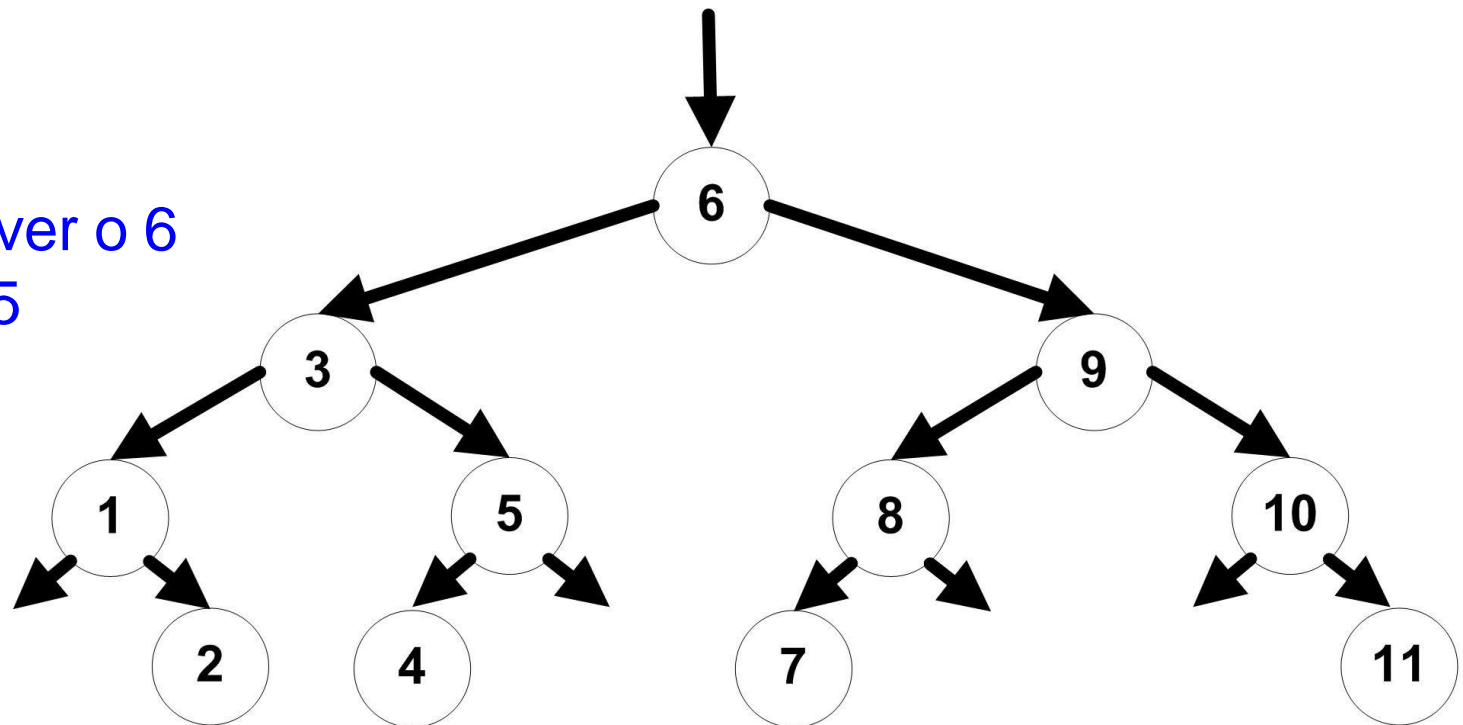
(3) Senão, se o elemento estiver em um **nó intermediário com dois filhos**, o elemento a ser removido deve ser substituído ou pelo **maior nó da subárvore à esquerda** ou **menor nó da subárvore à direita**



# Funcionamento Básico da Remoção

(3) Senão, se o elemento estiver em um **nó intermediário com dois filhos**, o elemento a ser removido deve ser substituído ou pelo **maior nó da subárvore à esquerda** ou **menor nó da subárvore à direita**

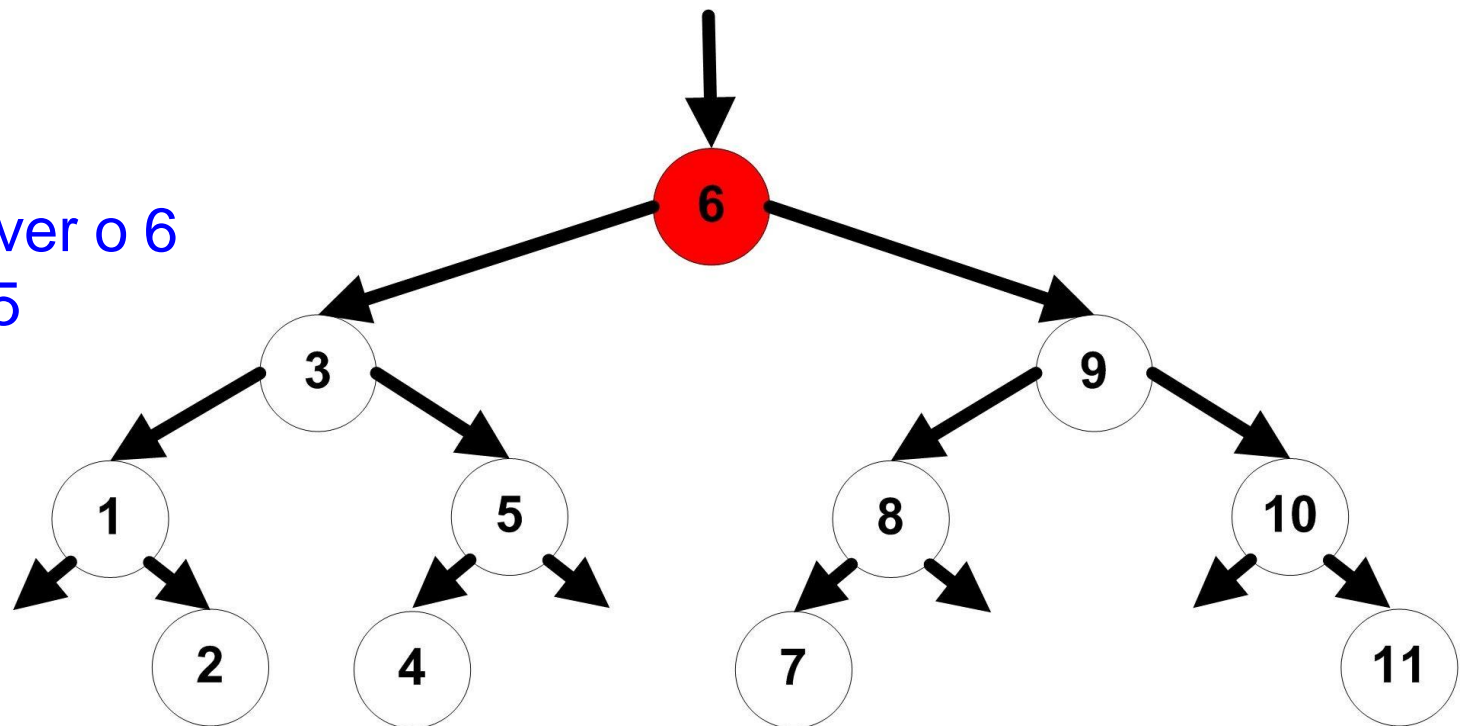
Exemplo: Remover o 6 e substituir pelo 5



# Funcionamento Básico da Remoção

(3) Senão, se o elemento estiver em um **nó intermediário com dois filhos**, o elemento a ser removido deve ser substituído ou pelo **maior nó da subárvore à esquerda** ou **menor nó da subárvore à direita**

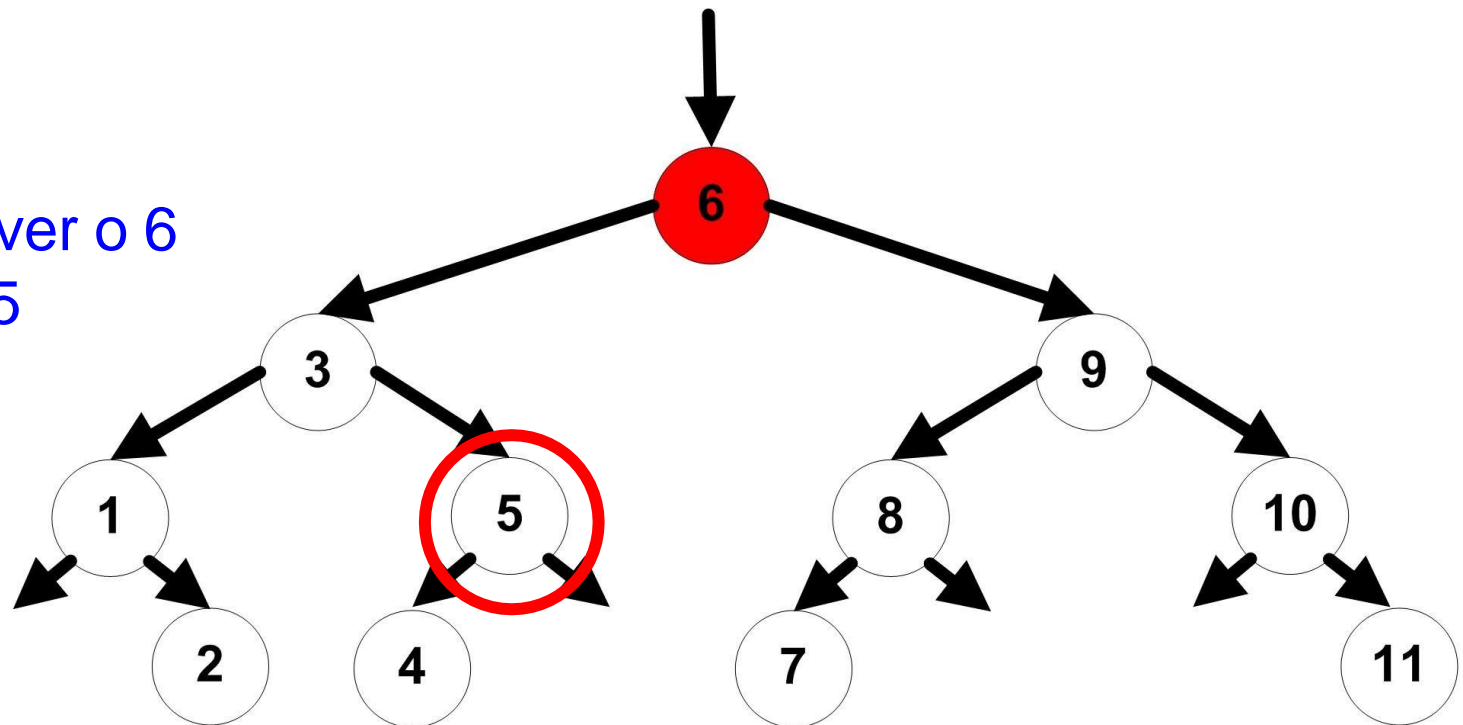
Exemplo: Remover o 6 e substituir pelo 5



# Funcionamento Básico da Remoção

(3) Senão, se o elemento estiver em um **nó intermediário com dois filhos**, o elemento a ser removido deve ser substituído ou pelo **maior nó da subárvore à esquerda** ou **menor nó da subárvore à direita**

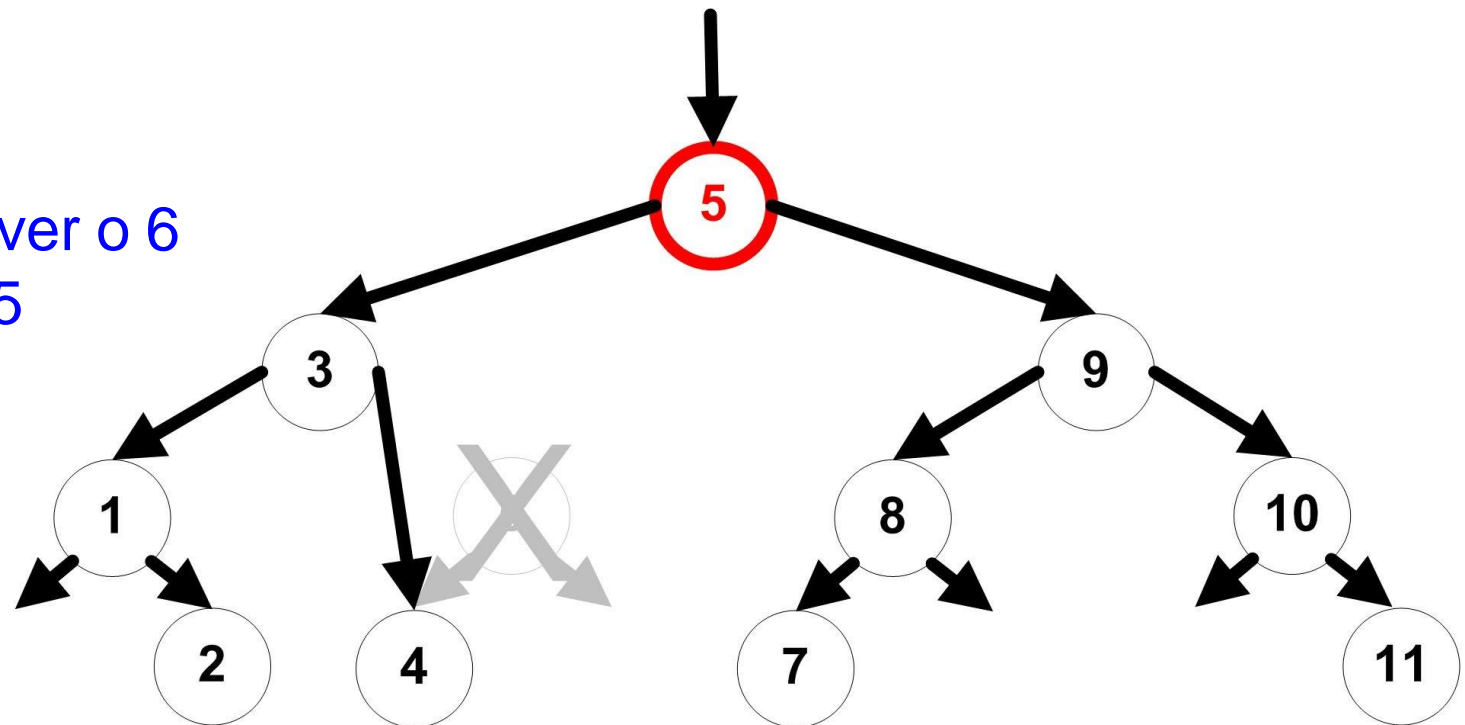
Exemplo: Remover o 6 e substituir pelo 5



# Funcionamento Básico da Remoção

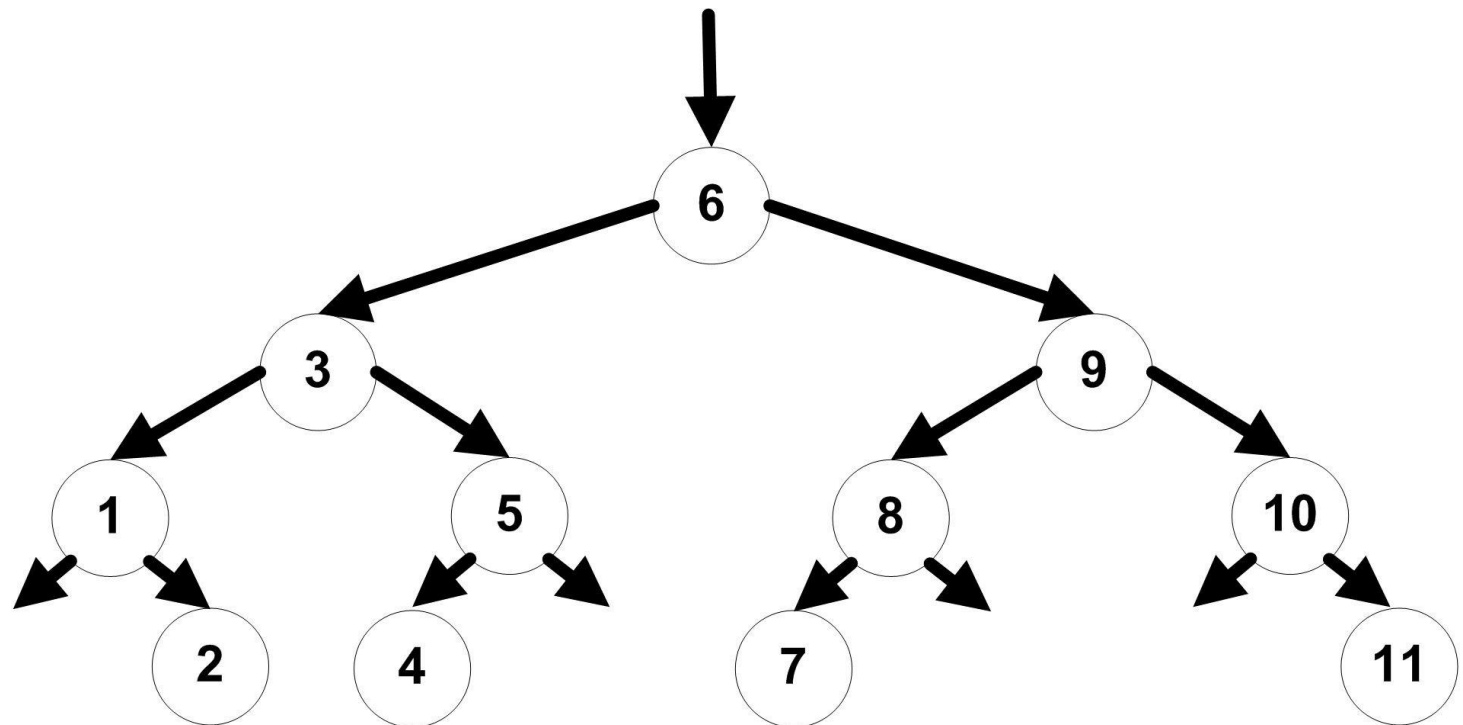
(3) Senão, se o elemento estiver em um **nó intermediário com dois filhos**, o elemento a ser removido deve ser substituído ou pelo **maior nó da subárvore à esquerda** ou **menor nó da subárvore à direita**

Exemplo: Remover o 6 e substituir pelo 5



# Funcionamento Básico da Remoção

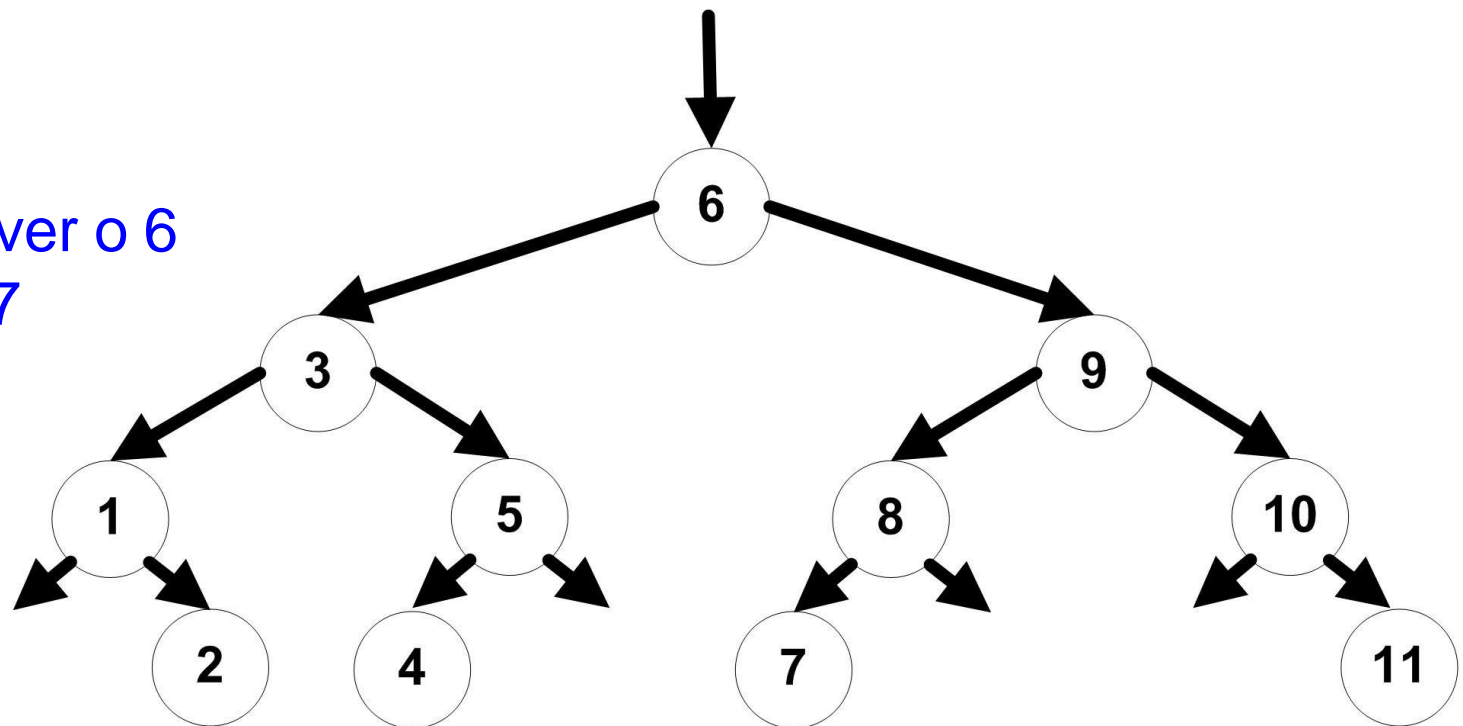
(3) Senão, se o elemento estiver em um **nó intermediário com dois filhos**, o elemento a ser removido deve ser substituído ou pelo **maior nó da subárvore à esquerda** ou **menor nó da subárvore à direita**



# Funcionamento Básico da Remoção

(3) Senão, se o elemento estiver em um **nó intermediário com dois filhos**, o elemento a ser removido deve ser substituído ou pelo **maior nó da subárvore à esquerda** ou **menor nó da subárvore à direita**

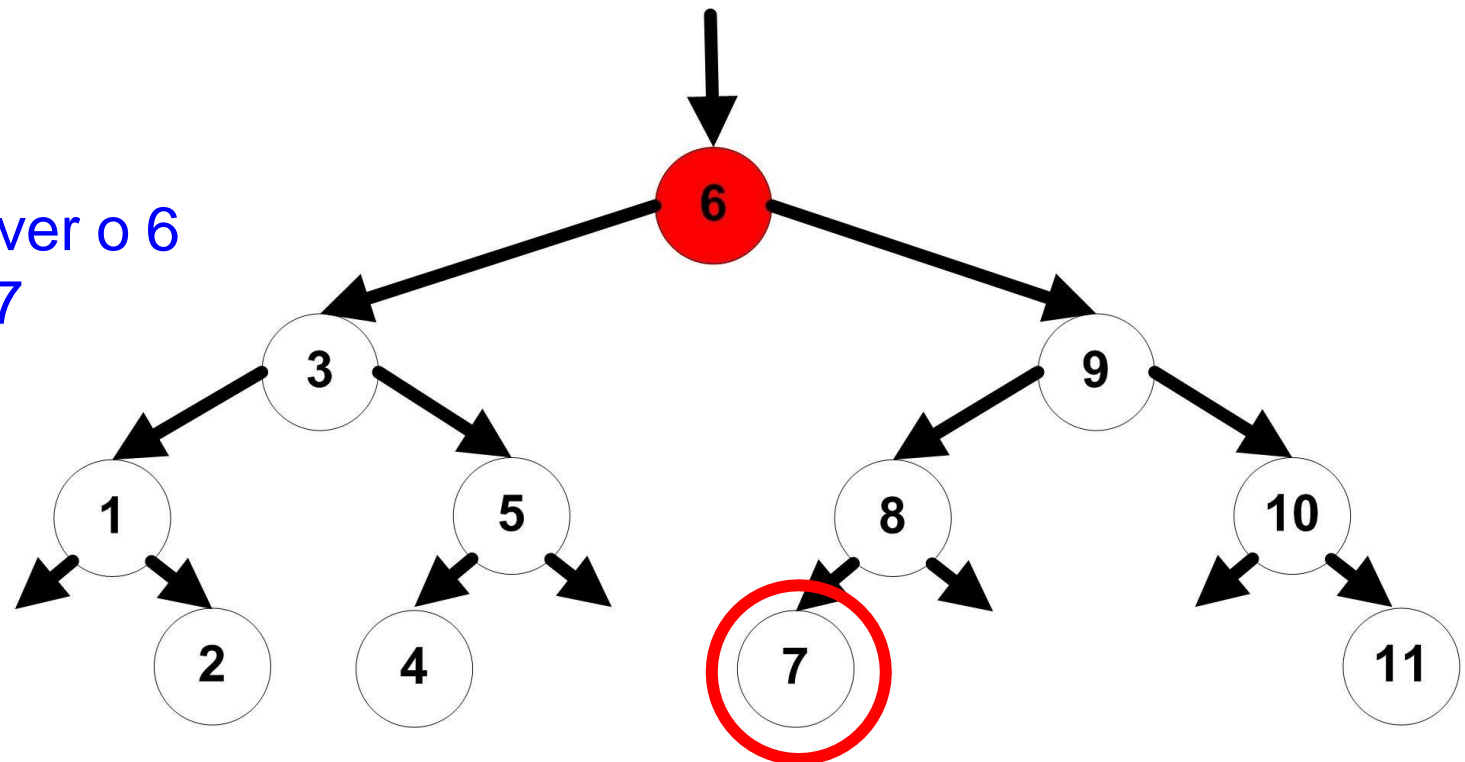
Exemplo: Remover o 6 e substituir pelo 7



# Funcionamento Básico da Remoção

(3) Senão, se o elemento estiver em um **nó intermediário com dois filhos**, o elemento a ser removido deve ser substituído ou pelo **maior nó da subárvore à esquerda** ou **menor nó da subárvore à direita**

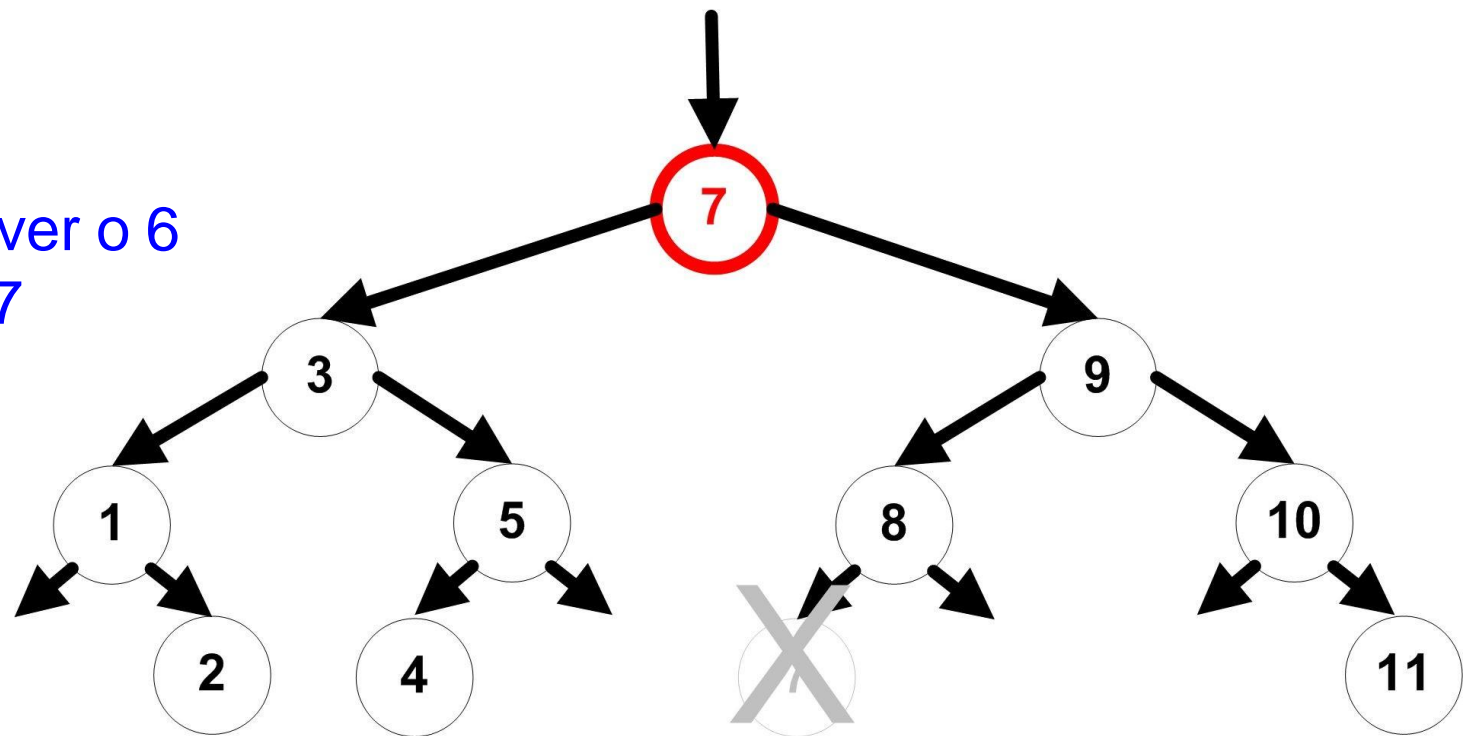
Exemplo: Remover o 6 e substituir pelo 7



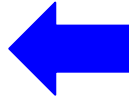
# Funcionamento Básico da Remoção

(3) Senão, se o elemento estiver em um **nó intermediário com dois filhos**, o elemento a ser removido deve ser substituído ou pelo **maior nó da subárvore à esquerda** ou **menor nó da subárvore à direita**

Exemplo: Remover o 6 e substituir pelo 7





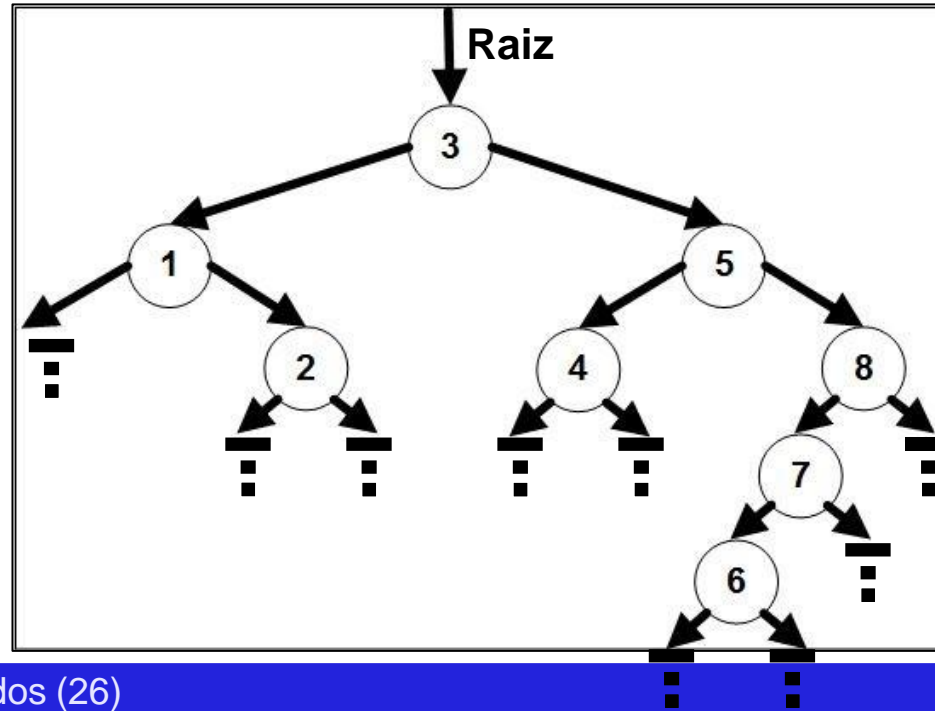
- Funcionamento básico
- **Algoritmo** 
- Análise de complexidade

# Classe Árvore Binária: Remoção

```
class ArvoreBinaria
{
    private No raiz;
    public ArvoreBinaria(){raiz = null;}
    public void Inserir(int x) { }
    public bool Pesquisar(int x) { }
    public void CaminharCentral() { }
    public void CaminharPre() { }
    public void CaminharPos() { }
    public void Remover(int x) { }
}
```

raiz

n(3)



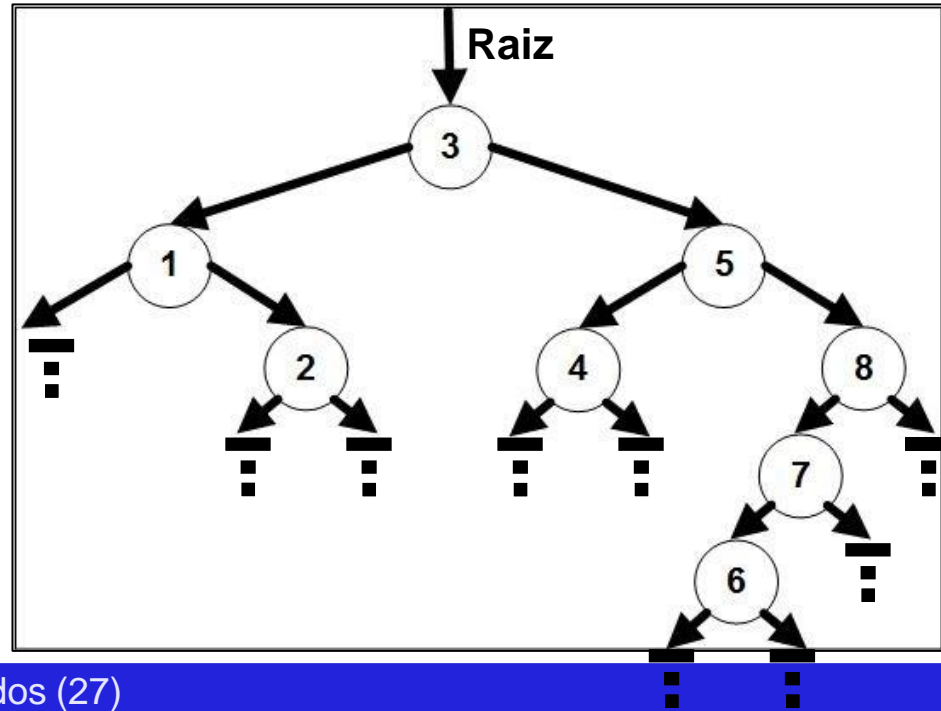
# Classe Árvore Binária: Remoção

```
class ArvoreBinaria
{
    private No raiz;
    public ArvoreBinaria(){raiz = null;}
    public void Inserir(int x) { }
    public bool Pesquisar(int x) { }
    public void CaminharCentral() { }
    public void CaminharPre() { }
    public void CaminharPos() { }
    public void Remover(int x) { }
}
```

raiz

n(3)

Vamos Remover o 2  
(uma folha) de nossa  
árvore



# Algoritmo de Remoção

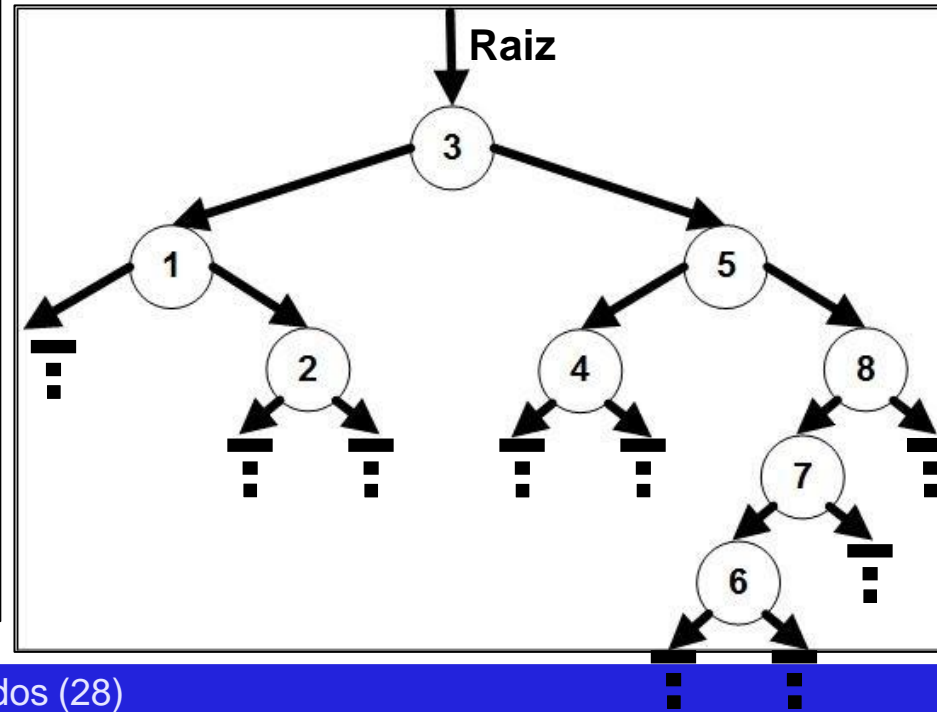
//Remover(2), folha

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}

No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```

raiz n(3) x 2



# Algoritmo de Remoção

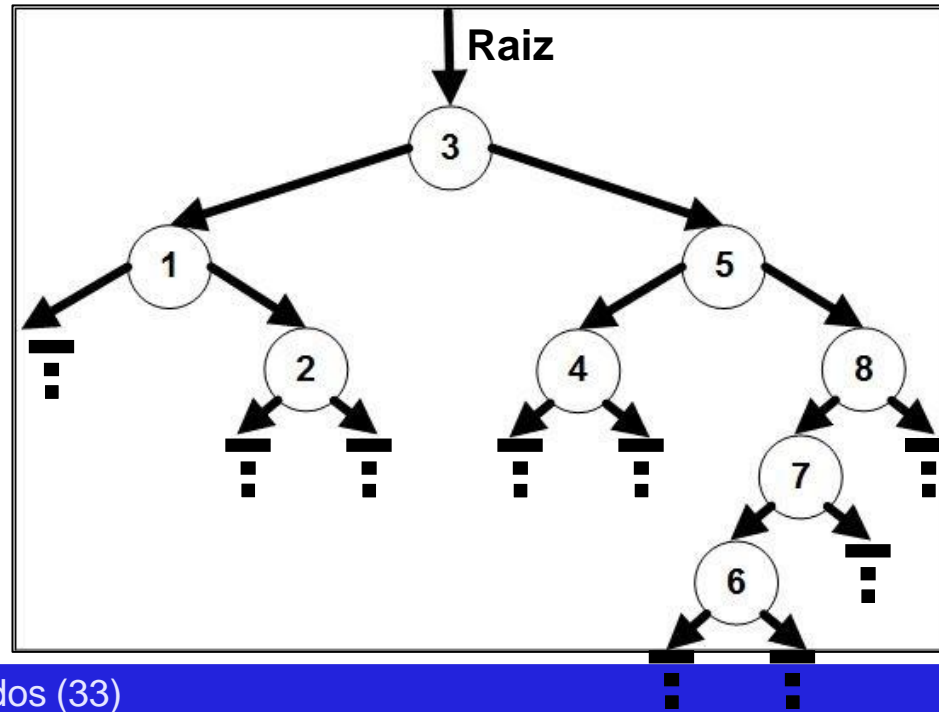
//Remover(2), folha

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}

No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```

raiz     $n(3)$     x    2



# Algoritmo de Remoção

//Remover(2), folha

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}
```

```
private No Remover(int x, No i) {
```

```
    if (i == null) {        throw new Exception("Erro!");
    } else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq);
    } else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir);
    } else if (i.Dir == null) {    i = i.Esq;
    } else if (i.Esq == null) {    i = i.Dir;
    } else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
```

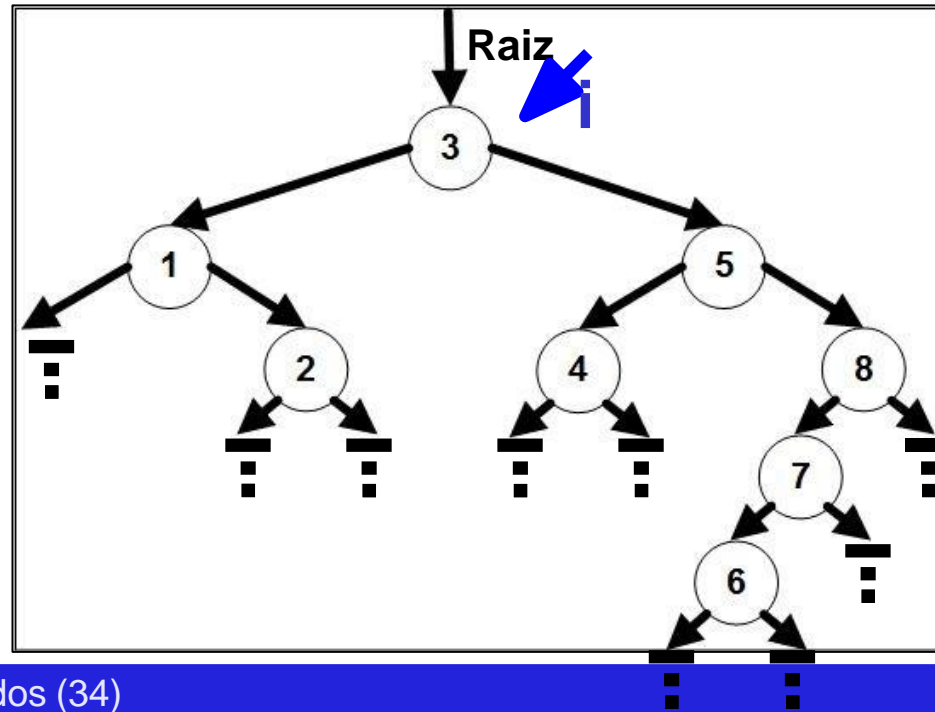
```
}
```

```
No MaiorEsq(No i, No j) {
```

```
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else {                j.Dir = MaiorEsq(i, j.Dir);        }
    return j;
```

```
}
```

raiz    n(3)    x    2    x    2    i    n(3)



# Algoritmo de Remoção

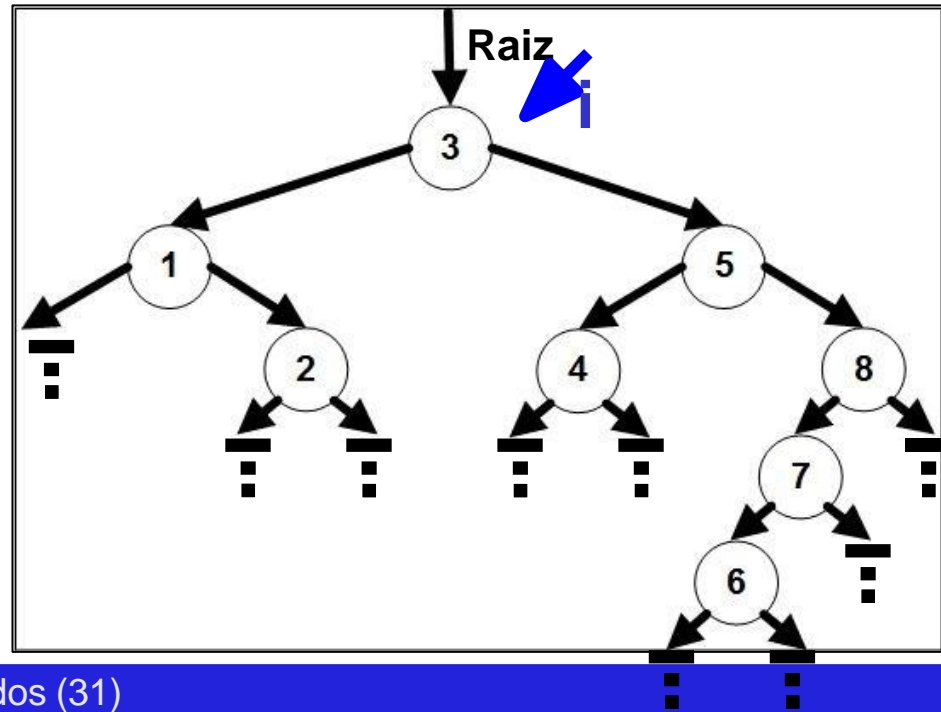
//Remover(2), folha

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!");
    } else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq);
    } else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir);
    } else if (i.Dir == null) { i = i.Esq;
    } else if (i.Esq == null) { i = i.Dir;
    } else { i.Esq = MaiorEsq(i, i.Esq);
    return i;
    }
    false: n(3) == null

No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento=j.Elemento; j=j.Esq; }
    else {
        j.Dir = MaiorEsq(i, j.Dir);
    }
    return j;
}
```

raiz    n(3)    x    2    x    2    i    n(3)



# Algoritmo de Remoção

//Remover(2), folha

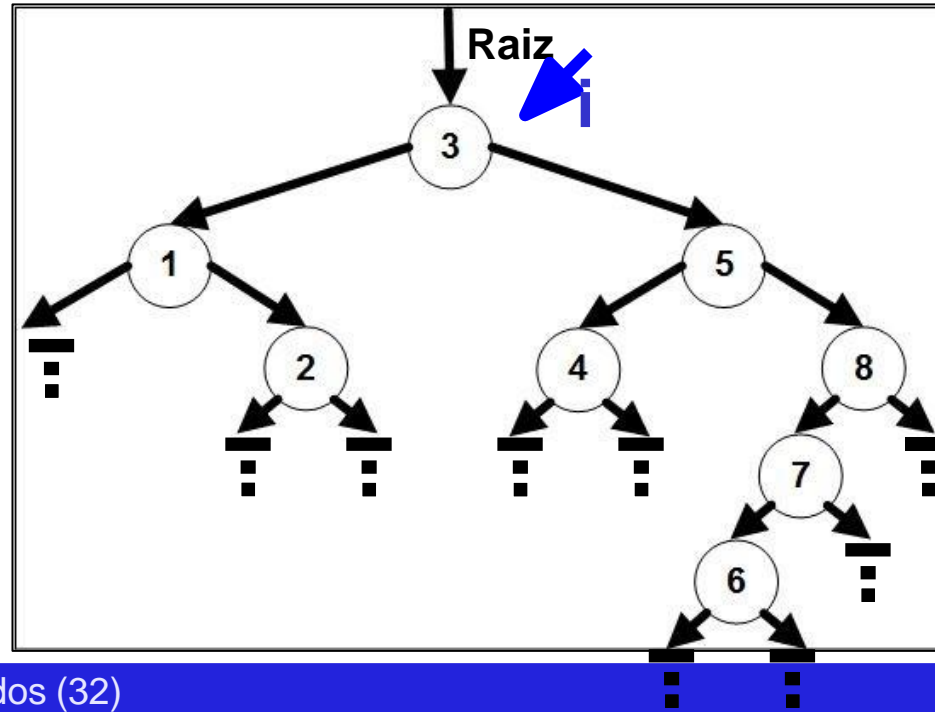
```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}

No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```

true: 2 < 3

raiz    n(3)    x    2    x    2    i    n(3)





# Algoritmo de Remoção

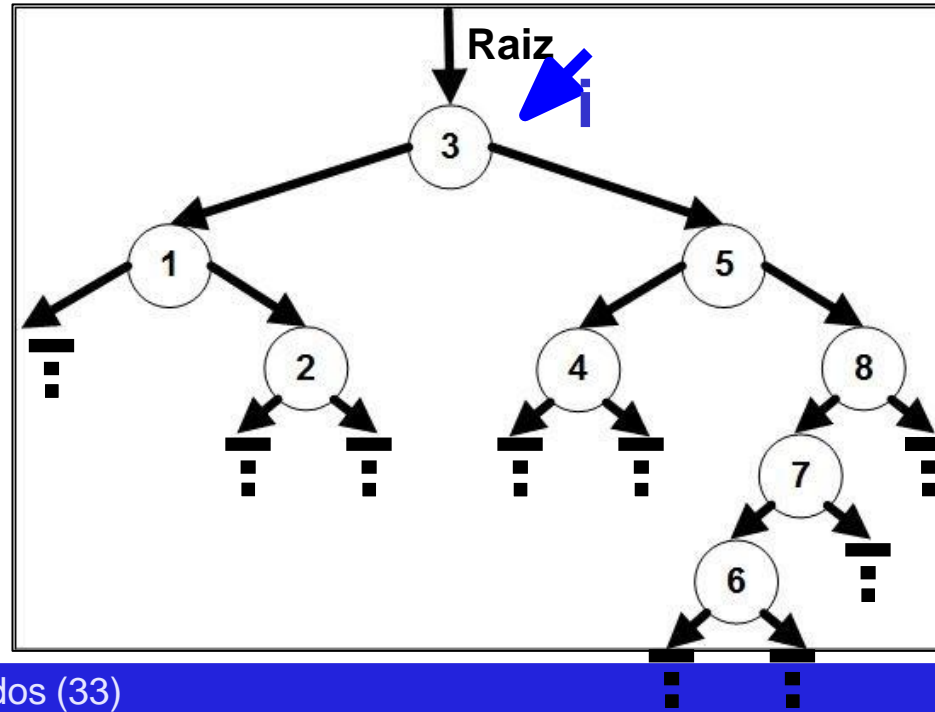
//Remover(2), folha

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}

No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```

raiz    n(3)    x    2    x    2    i    n(3)



# Algoritmo de Remoção

//Remover(2), folha

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}
```

```
private No Remover(int x, No i) {
```

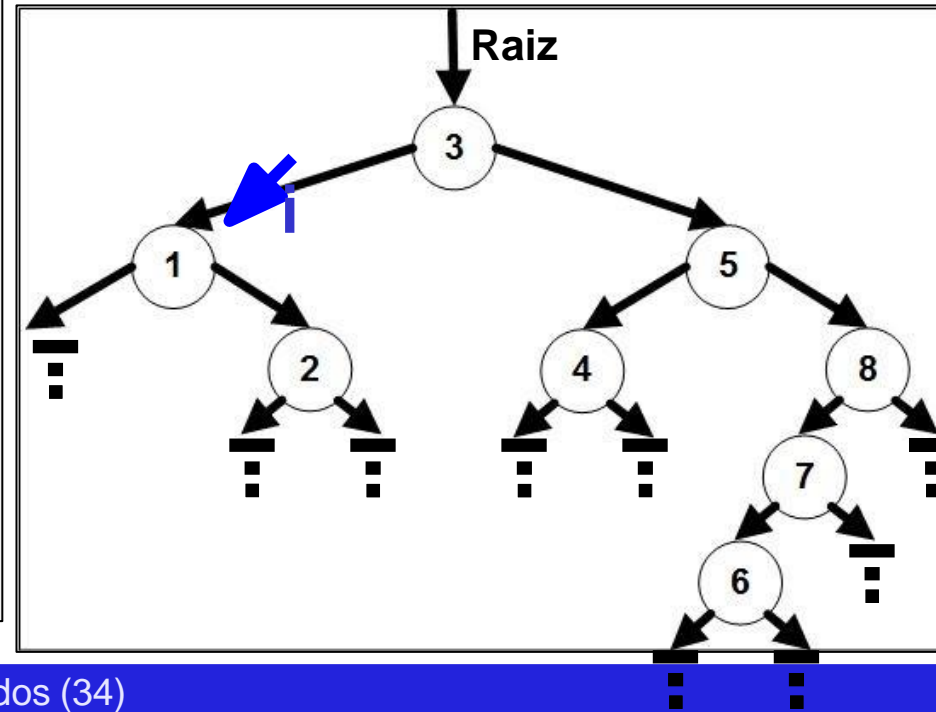
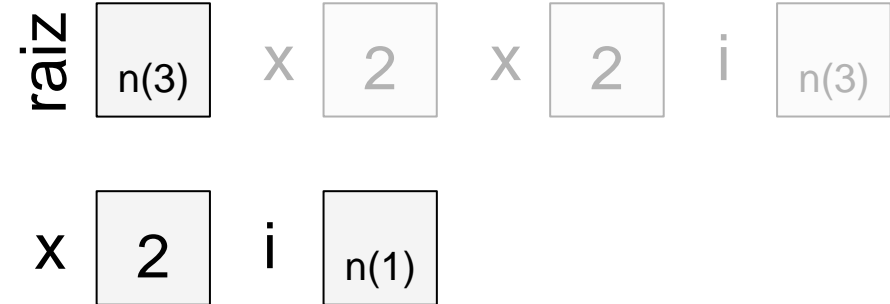
```
    if (i == null) {        throw new Exception("Erro!");
    } else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq);
    } else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir);
    } else if (i.Dir == null) {    i = i.Esq;
    } else if (i.Esq == null) {    i = i.Dir;
    } else {    i.Esq = MaiorEsq(i, i.Esq); }
    return i;
```

```
}
```

```
No MaiorEsq(No i, No j) {
```

```
    if (j.Dir == null) { i.Elemento=j.Elemento; j=j.Esq; }
    else {                j.Dir = MaiorEsq(i, j.Dir);        }
    return j;
```

```
}
```



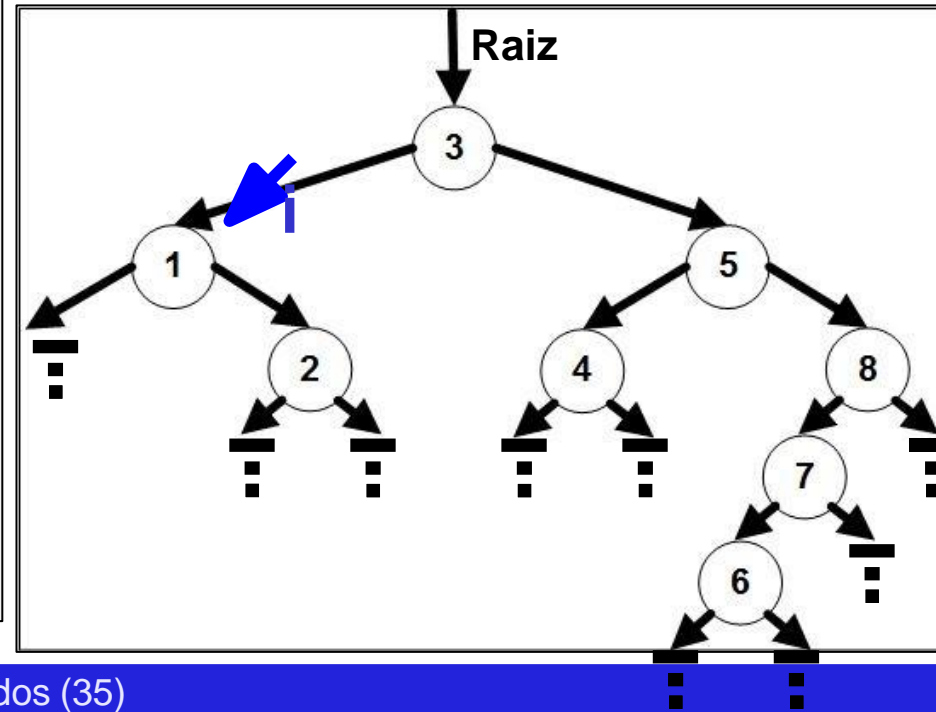
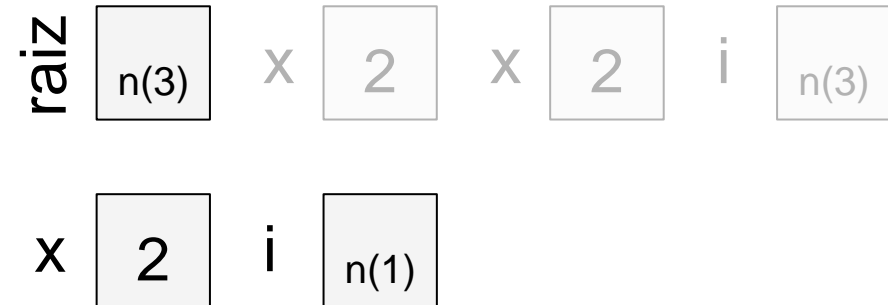
# Algoritmo de Remoção

//Remover(2), folha

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!");
    } else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq);
    } else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir);
    } else if (i.Dir == null) { i = i.Esq;
    } else if (i.Esq == null) { i = i.Dir;
    } else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}

No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```



# Algoritmo de Remoção

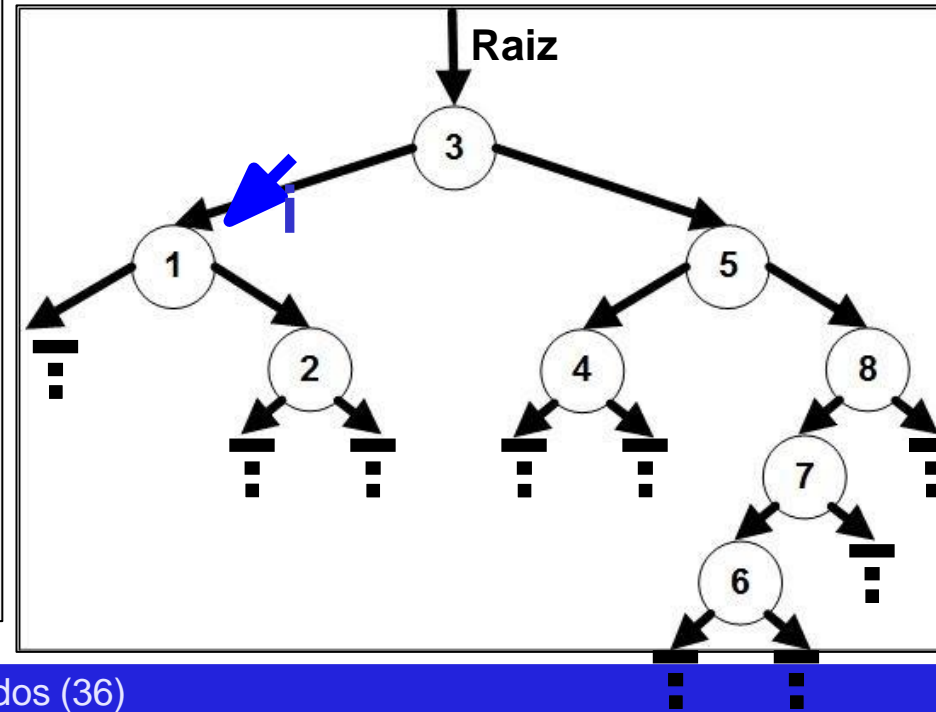
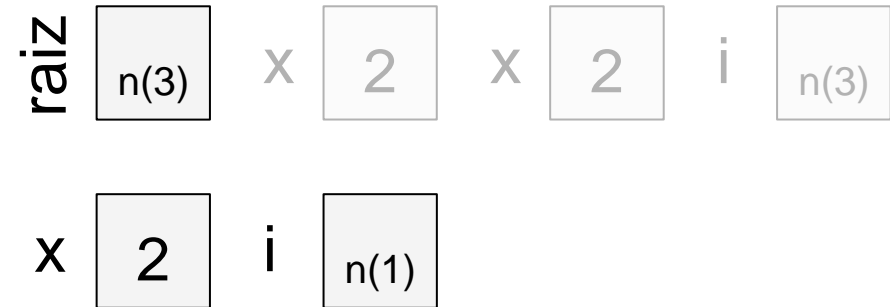
//Remover(2), folha

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}

No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```

false:  $2 < 1$



# Algoritmo de Remoção

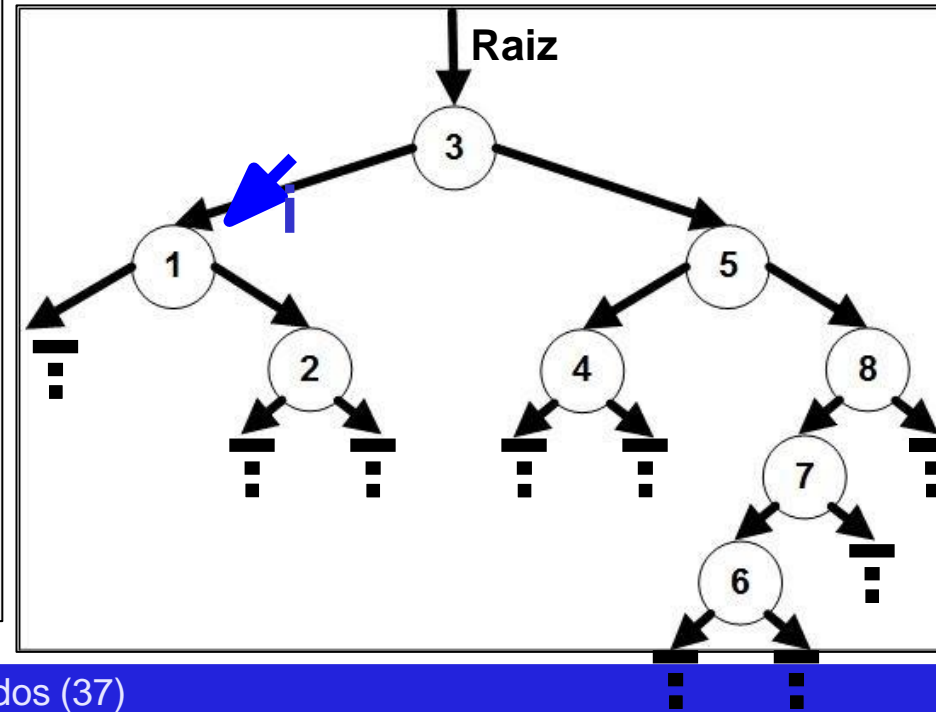
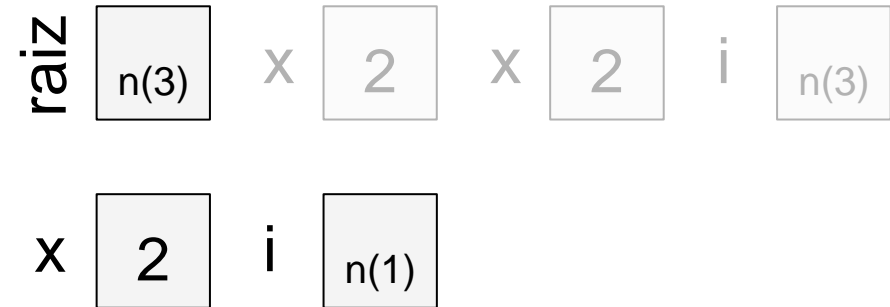
//Remover(2), folha

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}

No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```

true: 2 > 1



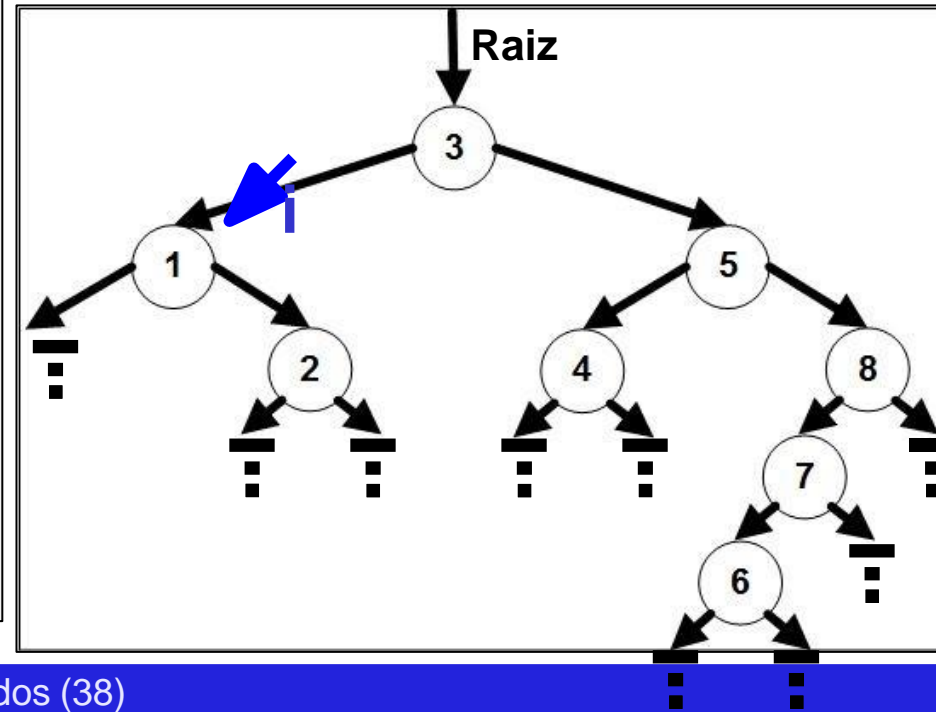
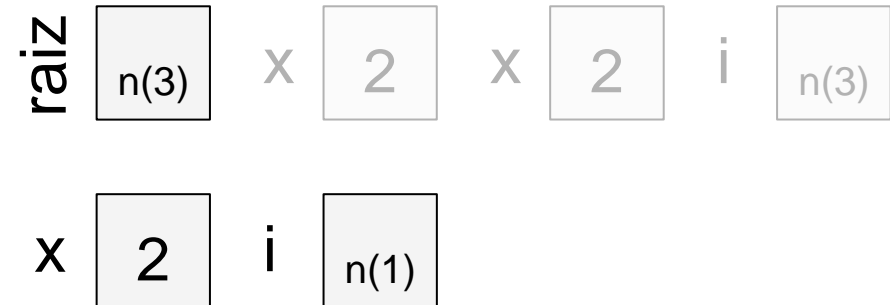
# Algoritmo de Remoção

//Remover(2), folha

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}

No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```



# Algoritmo de Remoção

```
//Remover(2), folha
```

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}
```

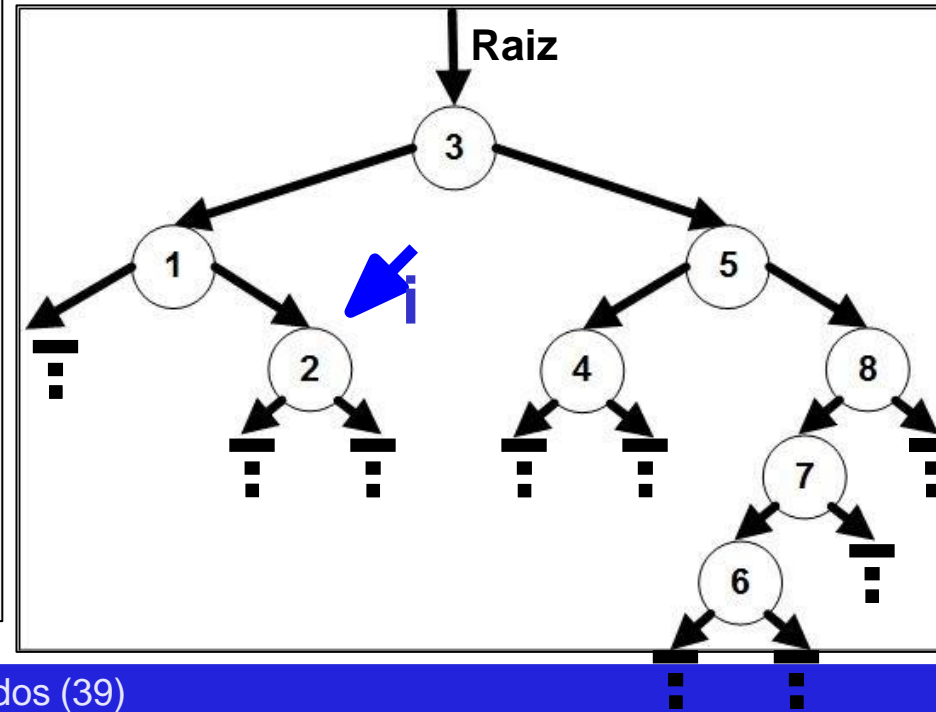
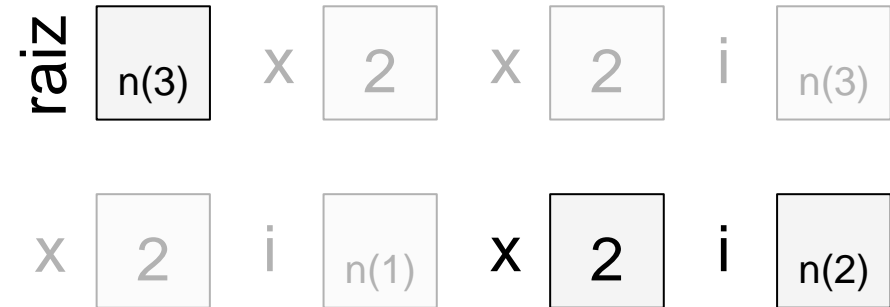
```
private No Remover(int x, No i) {
```

```

if (i == null) { throw new Exception("Erro!");
} else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq);
} else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir);
} else if (i.Dir == null) { i = i.Esq;
} else if (i.Esq == null) { i = i.Dir;
} else { i.Esq = MaiorEsq(i, i.Esq); }
return i;
}

```

```
No MaiorEsq(No i, No j) {
    if (j.Dir == null){ i.Elemento=j.Elemento; j=j.Esq; }
    else {                j.Dir = MaiorEsq(i, j.Dir);        }
    return j;
}
```



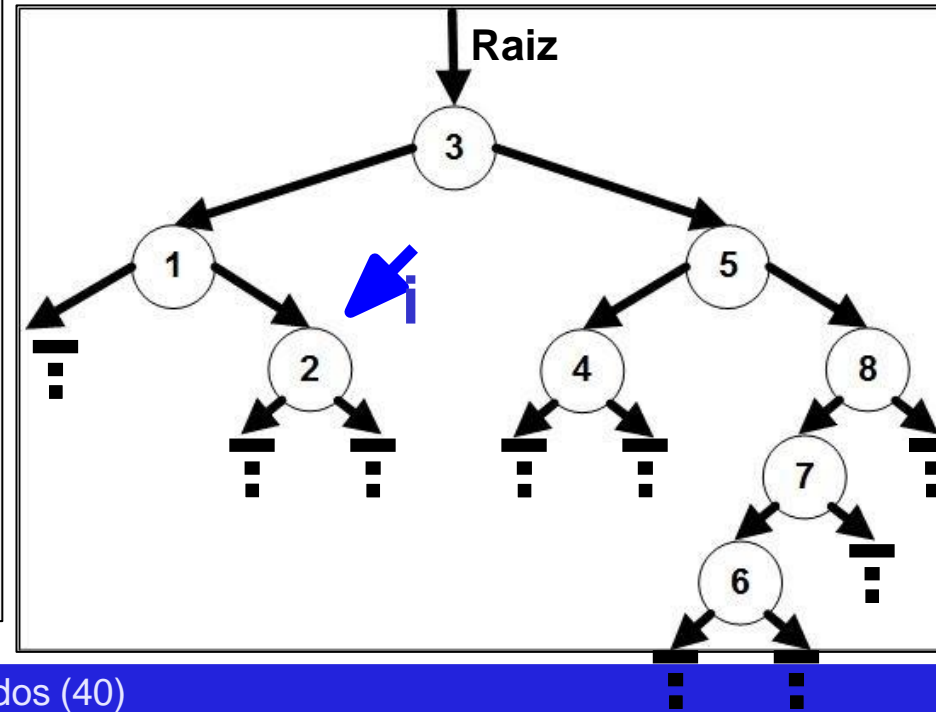
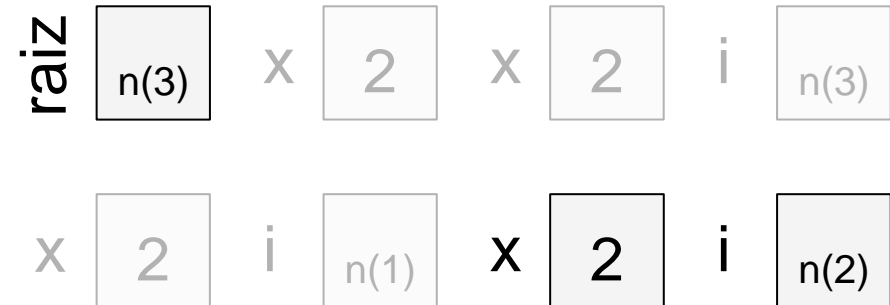
# Algoritmo de Remoção

//Remover(2), folha

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!");
    } else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq);
    } else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir);
    } else if (i.Dir == null) { i = i.Esq;
    } else if (i.Esq == null) { i = i.Dir;
    } else { i.Esq = MaiorEsq(i, i.Esq);
    return i;
    }
    false: n(2) == null

No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento=j.Elemento; j=j.Esq; }
    else {
        j.Dir = MaiorEsq(i, j.Dir);
    }
    return j;
}
```





# Algoritmo de Remoção

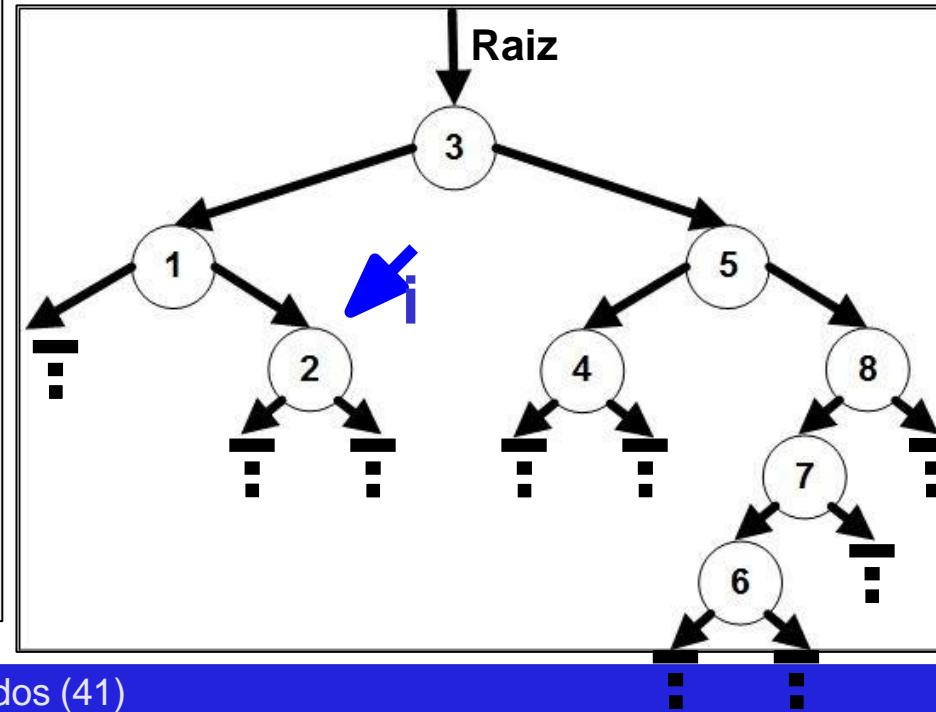
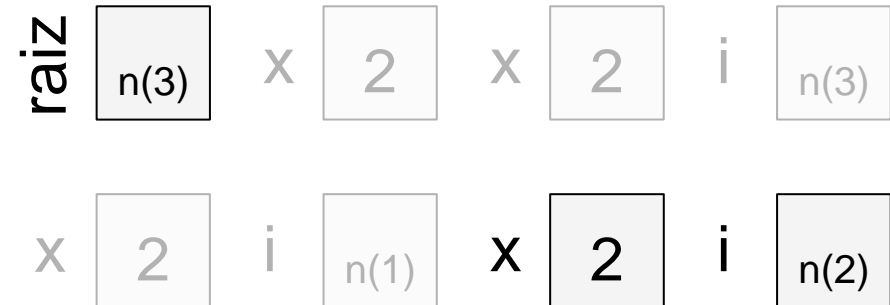
//Remover(2), folha

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}

No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```

false:  $2 < 2$



# Algoritmo de Remoção

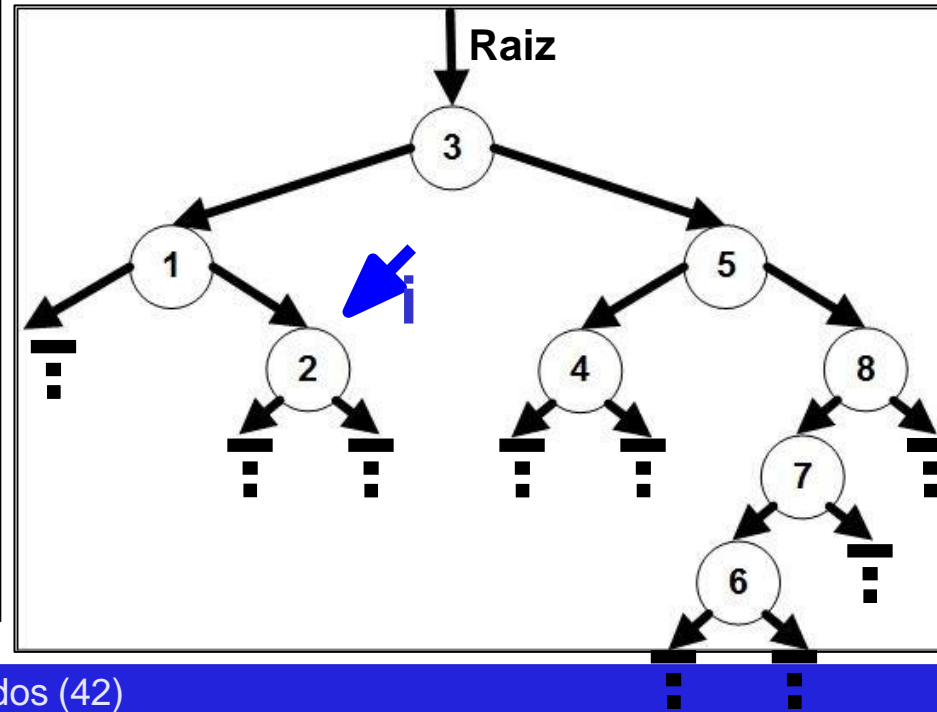
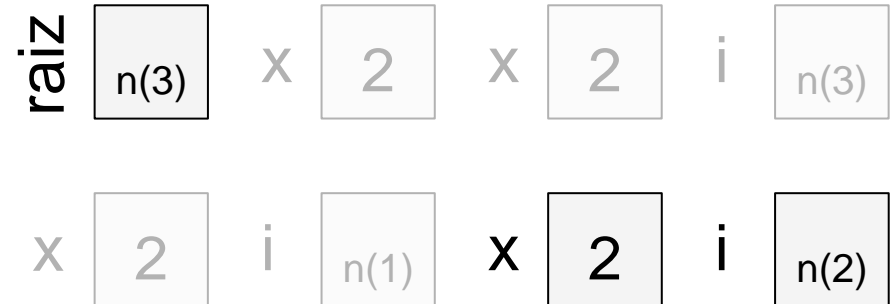
//Remover(2), folha

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}

No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```

false: 2 > 2



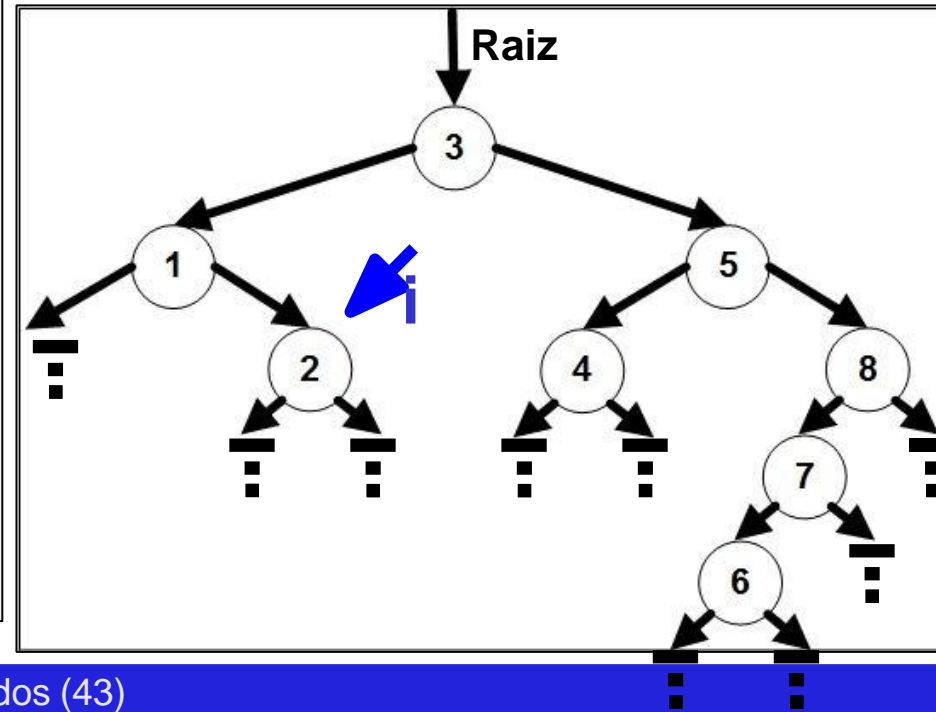
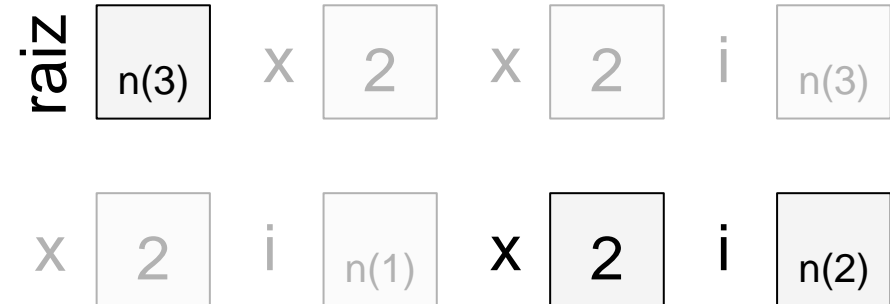
# Algoritmo de Remoção

//Remover(2), folha

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}

No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```



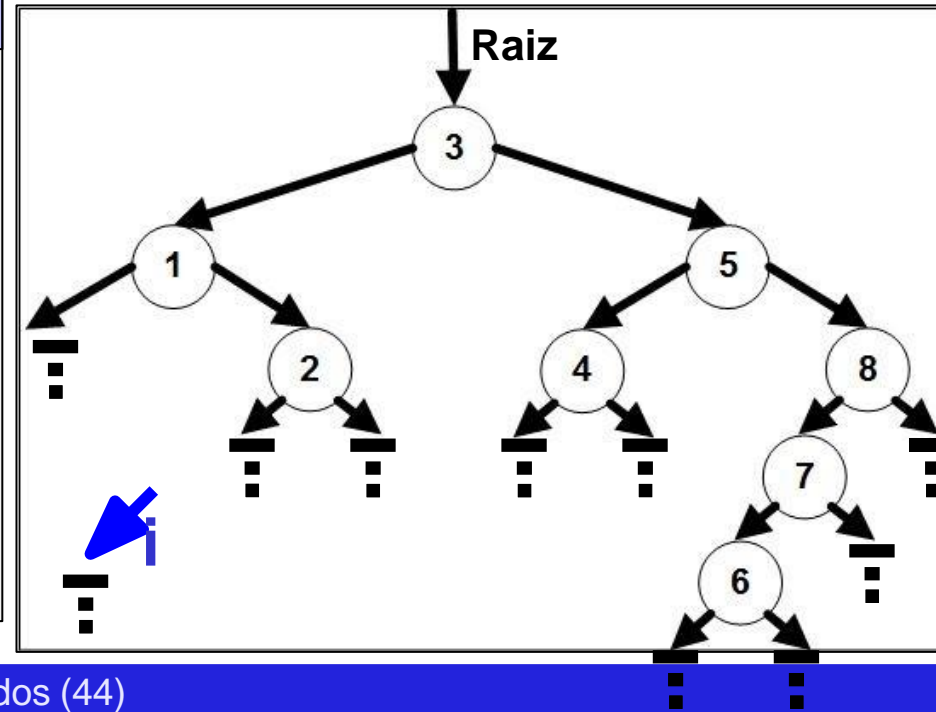
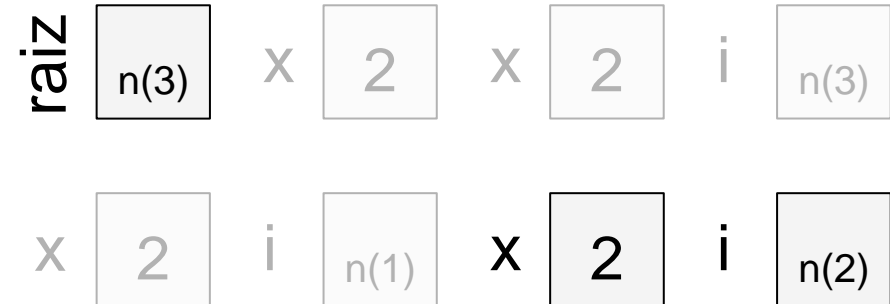
# Algoritmo de Remoção

//Remover(2), folha

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}

No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```



# Algoritmo de Remoção

//Remover(2), folha

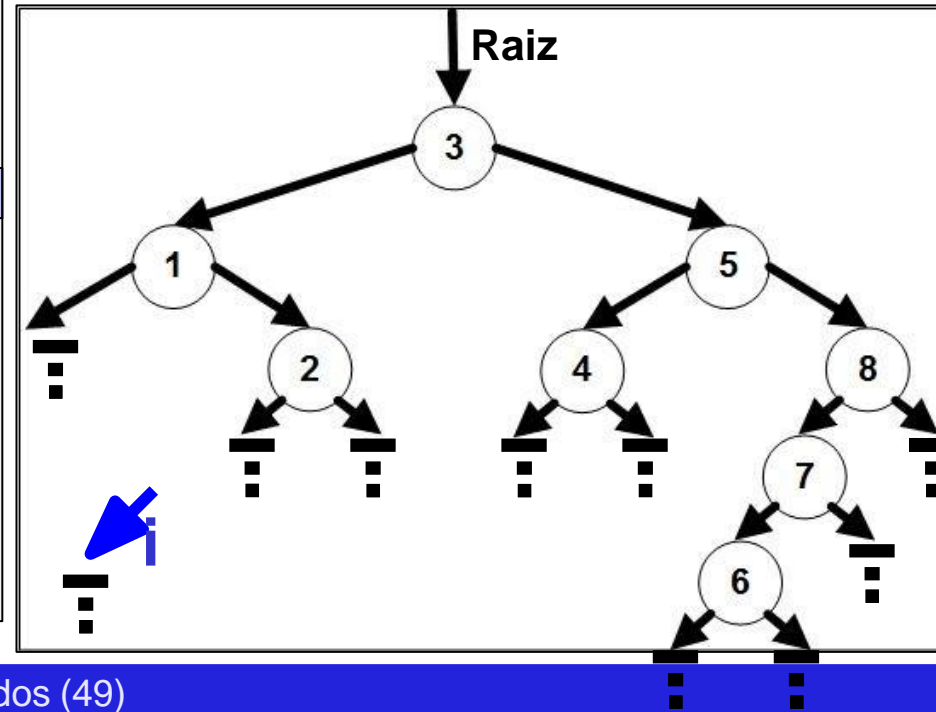
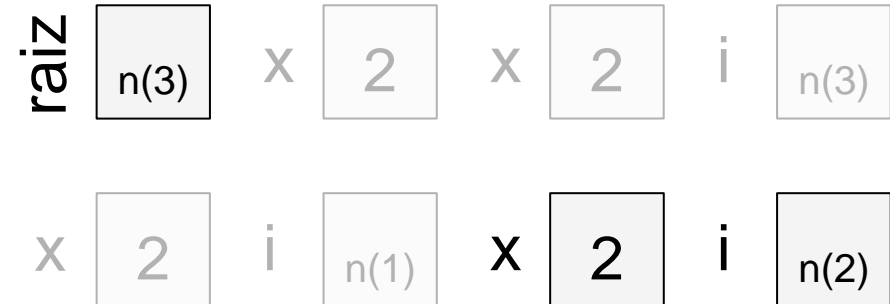
```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
}
```

return i;

Retorna null

```
No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```



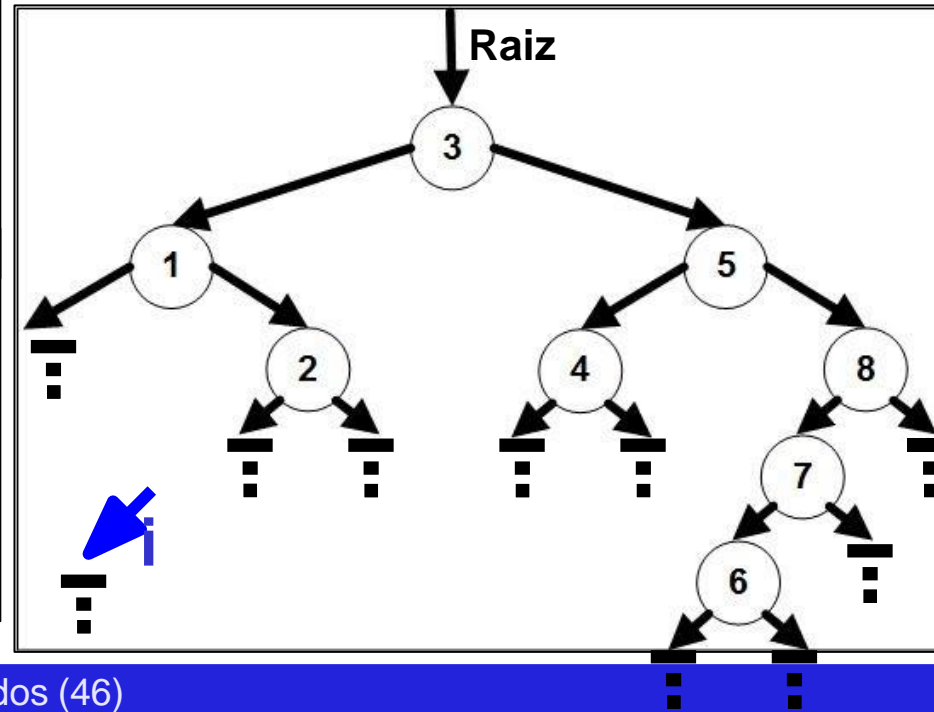
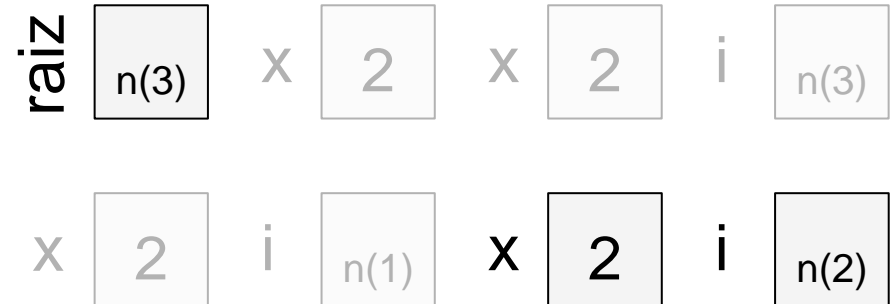
# Algoritmo de Remoção

//Remover(2), folha

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}
```

```
No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```



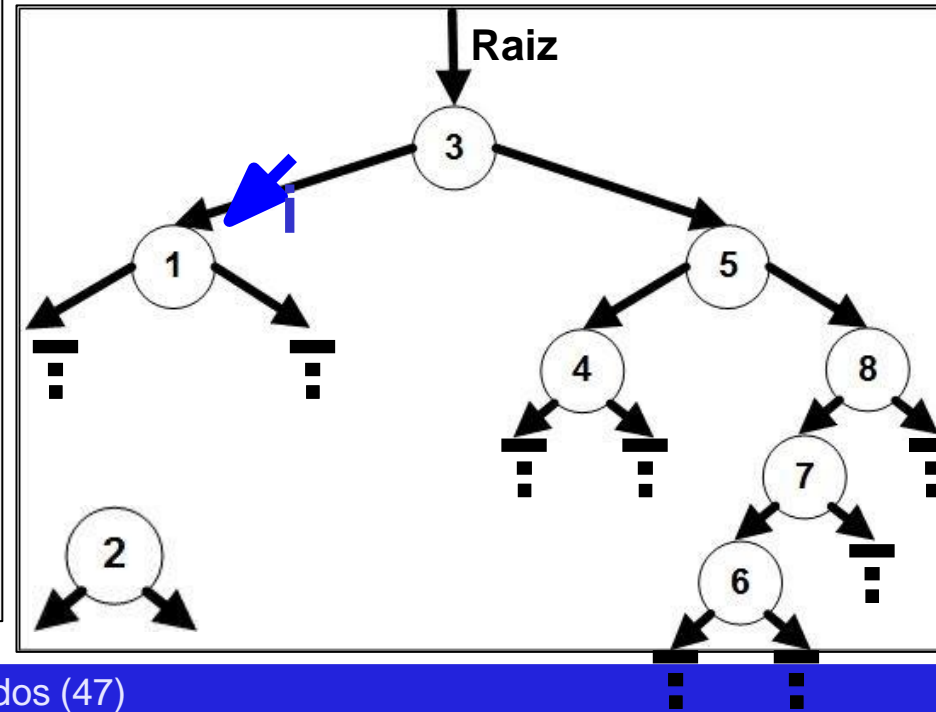
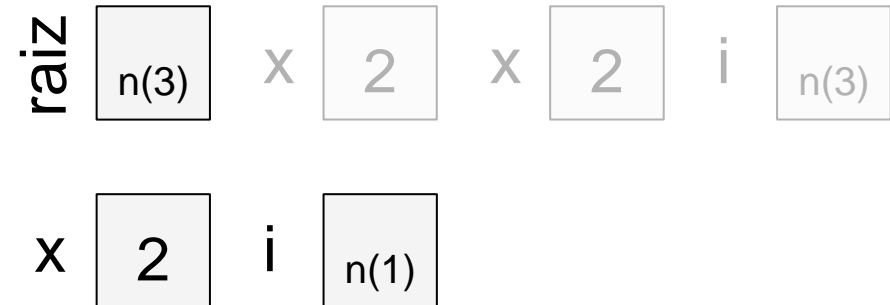
# Algoritmo de Remoção

//Remover(2), folha

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}

No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```



# Algoritmo de Remoção

//Remover(2), folha

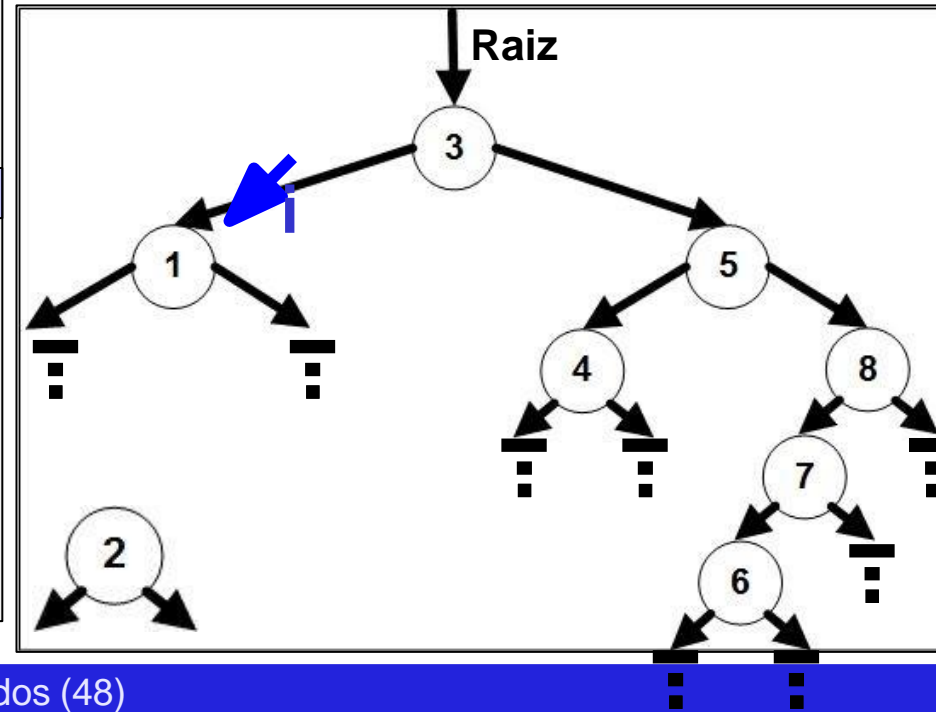
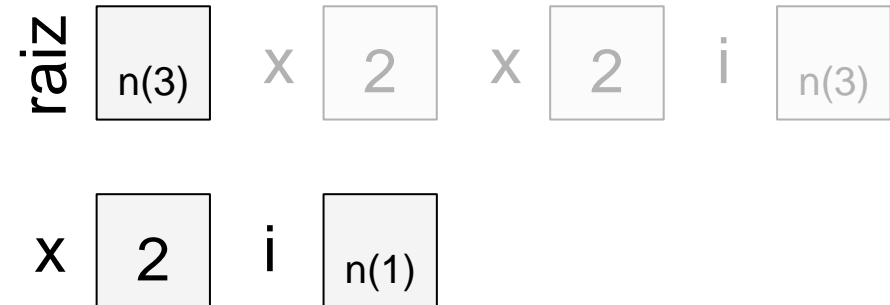
```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
}
```

return i;

Retorna n(1)

```
No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```





# Algoritmo de Remoção

//Remover(2), folha

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}
```

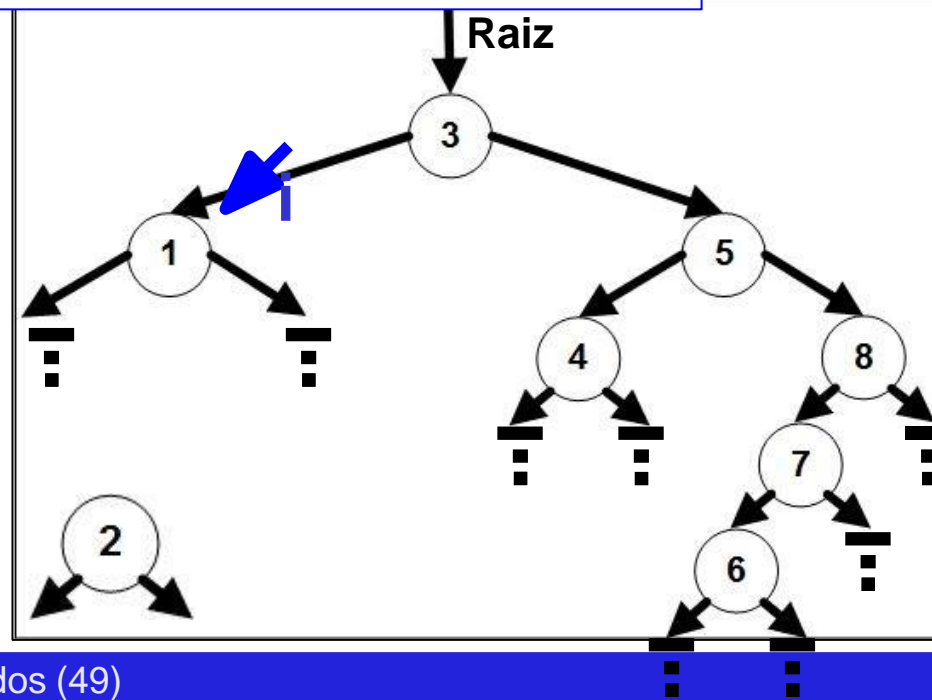
```
private int Remover(int x, No i) {
    if (i == null) return null;
    else if (i.Elemento == x) {
        if (i.Dir == null) {
            return null;
        } else if (i.Esq == null) {
            return i.Dir;
        } else {
            i.Esq = MaiorEsq(i, i.Esq);
        }
    }
    return i;
}
```

Retorna n(1)

```
No MaiorEsq(No i, No j) {
    if (j.Dir == null) {
        i.Elemento = j.Elemento;
        j = j.Esq;
    } else {
        j.Dir = MaiorEsq(i, j.Dir);
    }
    return j;
}
```

raiz n(3) x 2 x 2 i n(3)

Após a coleta de lixo  
(que não controlamos quando ela acontece)...



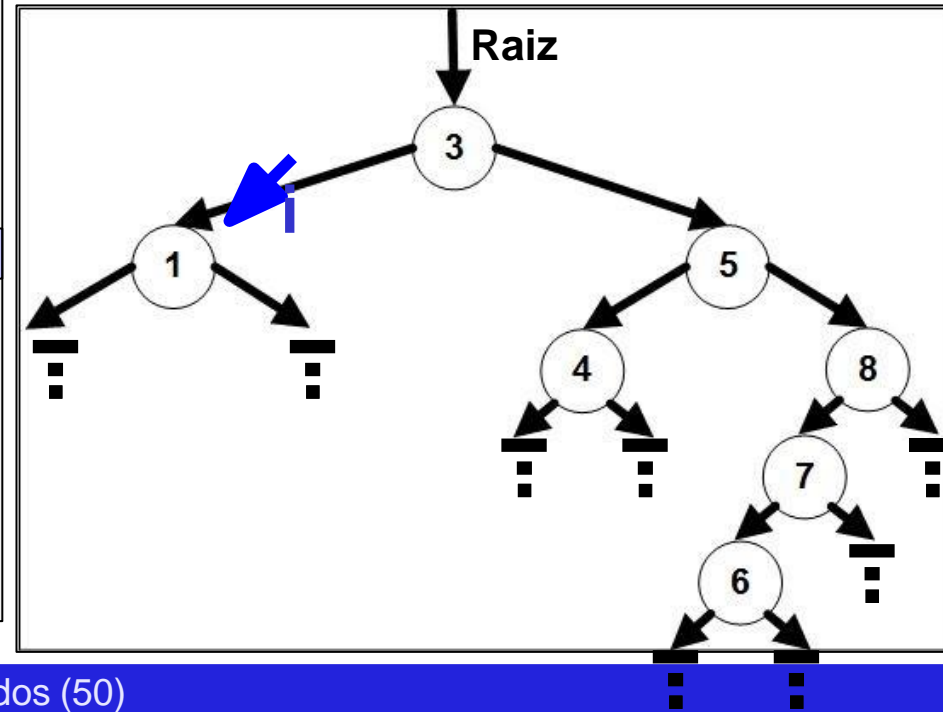
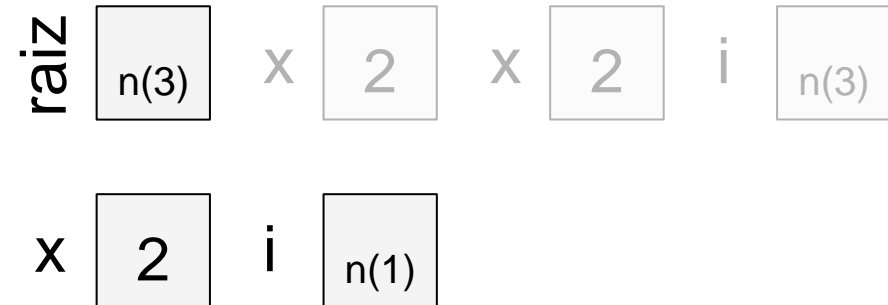
# Algoritmo de Remoção

//Remover(2), folha

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}
```

```
No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```



# Algoritmo de Remoção

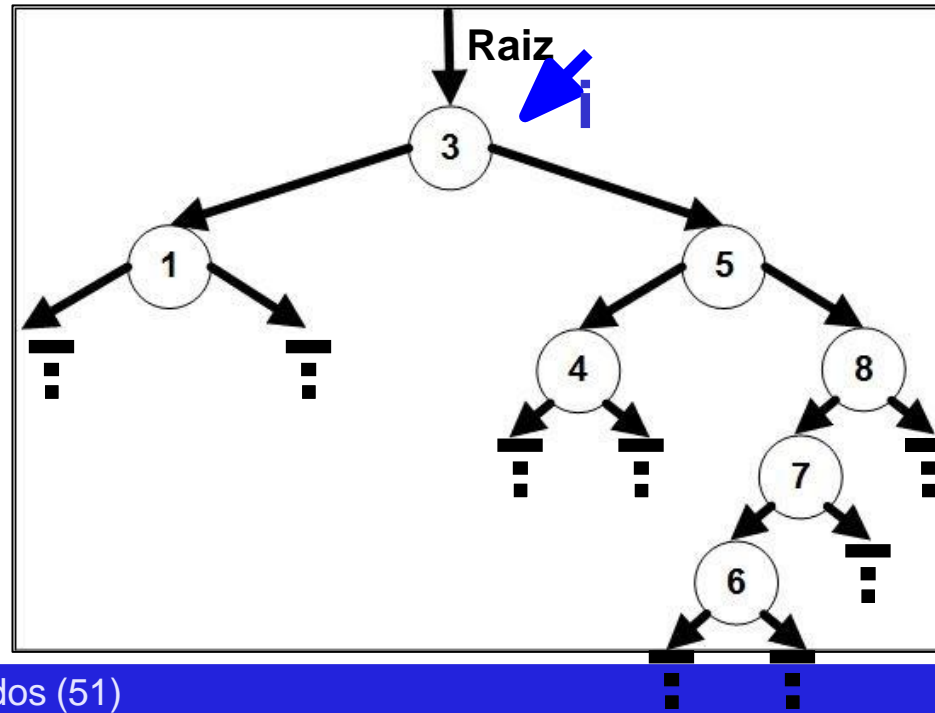
//Remover(2), folha

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}

No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```

raiz    n(3)    x    2    x    2    i    n(3)



# Algoritmo de Remoção

//Remover(2), folha

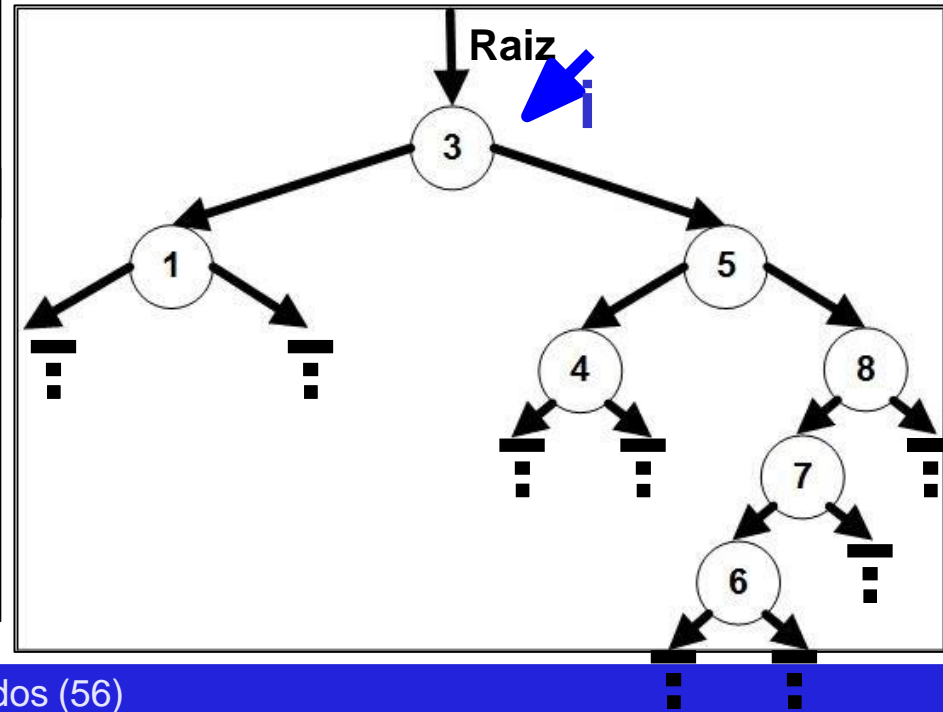
```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}
```

Retorna n(3)

```
No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```

raiz    n(3)    x    2    x    2    i    n(3)



# Algoritmo de Remoção

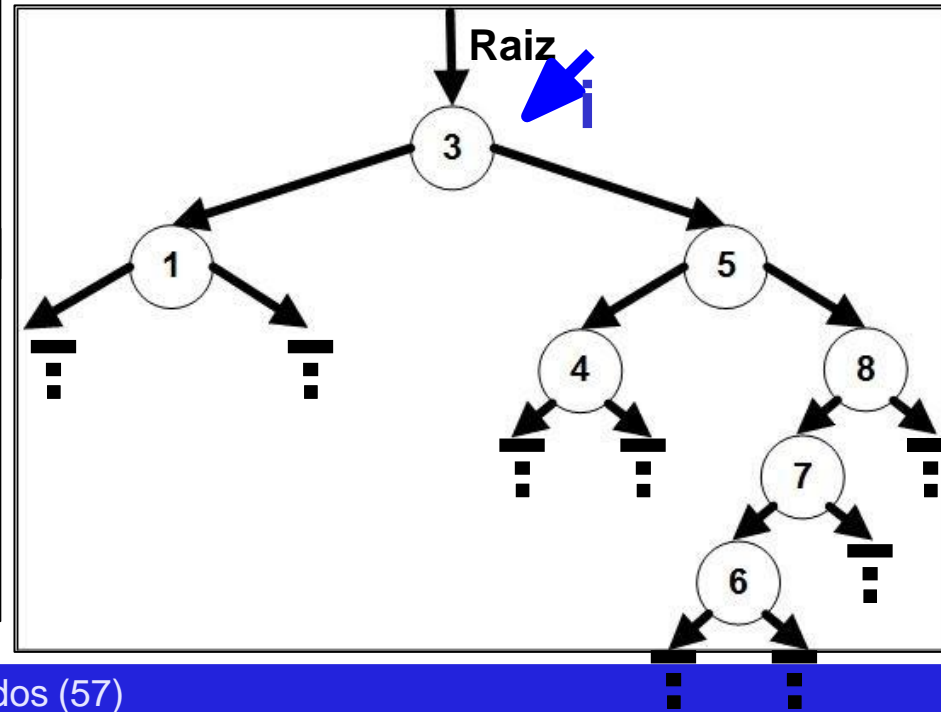
//Remover(2), folha

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}
```

```
No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```

raiz    n(3)    x    2    x    2    i    n(3)



# Algoritmo de Remoção

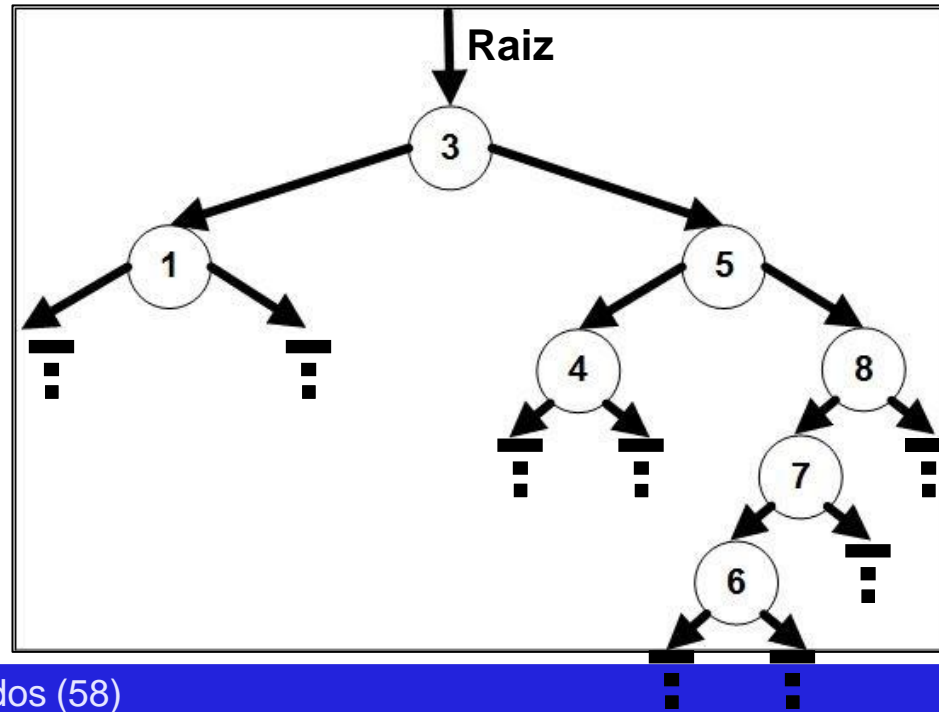
//Remover(2), folha

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}

No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```

raiz n(3) x 2



# Algoritmo de Remoção

//Remover(2), folha

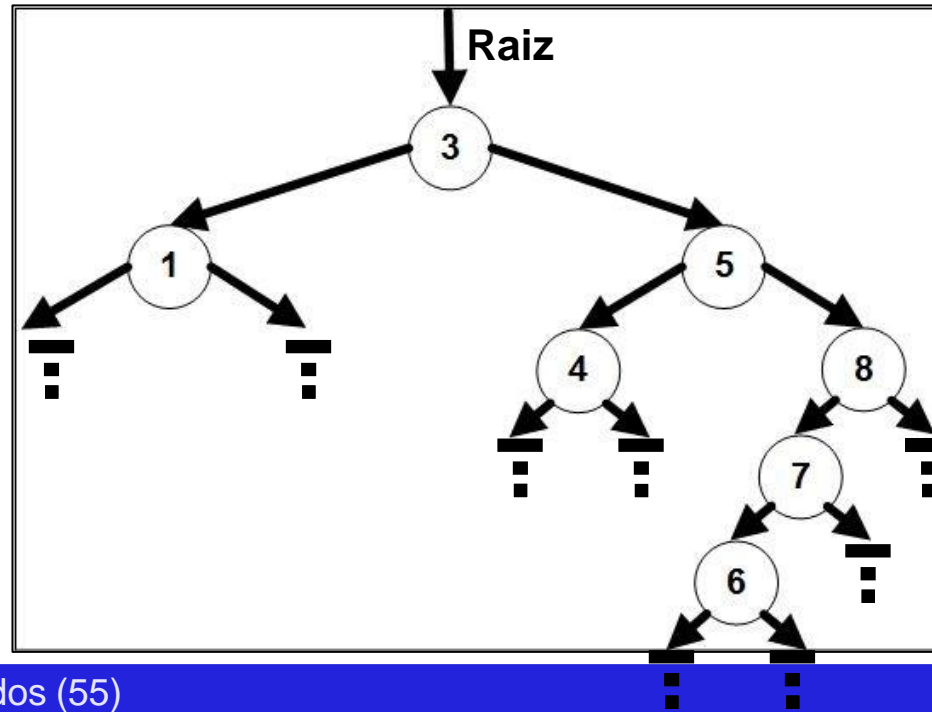
```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}
```

```
private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}
```

```
No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```

raiz

n(3)



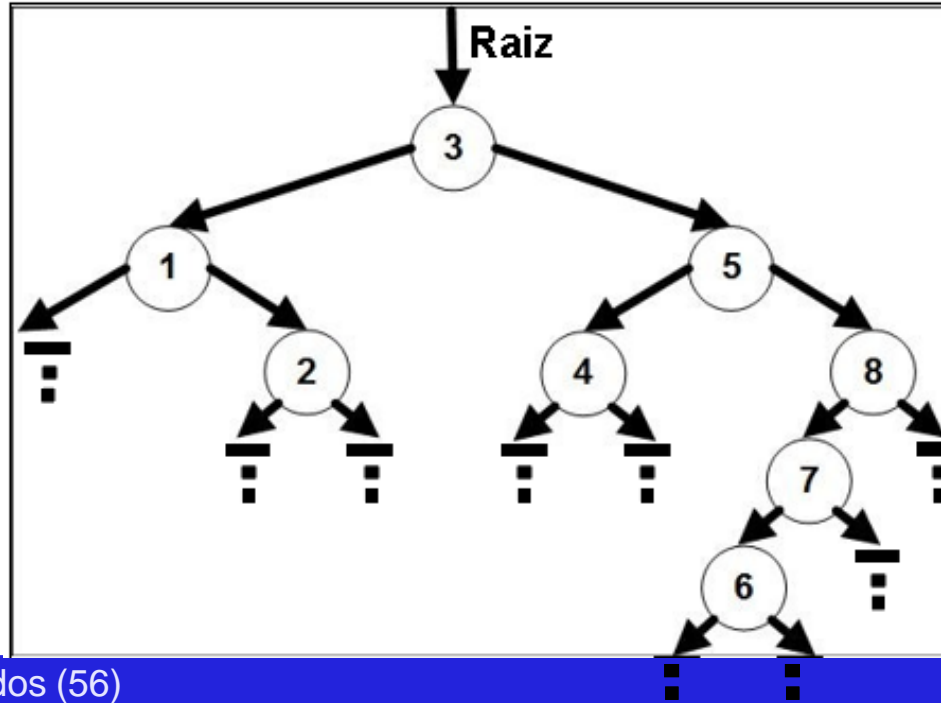
# Classe Árvore Binária: Remoção

```
class ArvoreBinaria
{
    private No raiz;
    public ArvoreBinaria(){raiz = null;}
    public void Inserir(int x) { }
    public bool Pesquisar(int x) { }
    public void CaminharCentral() { }
    public void CaminharPre() { }
    public void CaminharPos() { }
    public void Remover(int x) { }
}
```

Voltando com o 2 antes  
de fazer outra remoção

raiz

n(3)



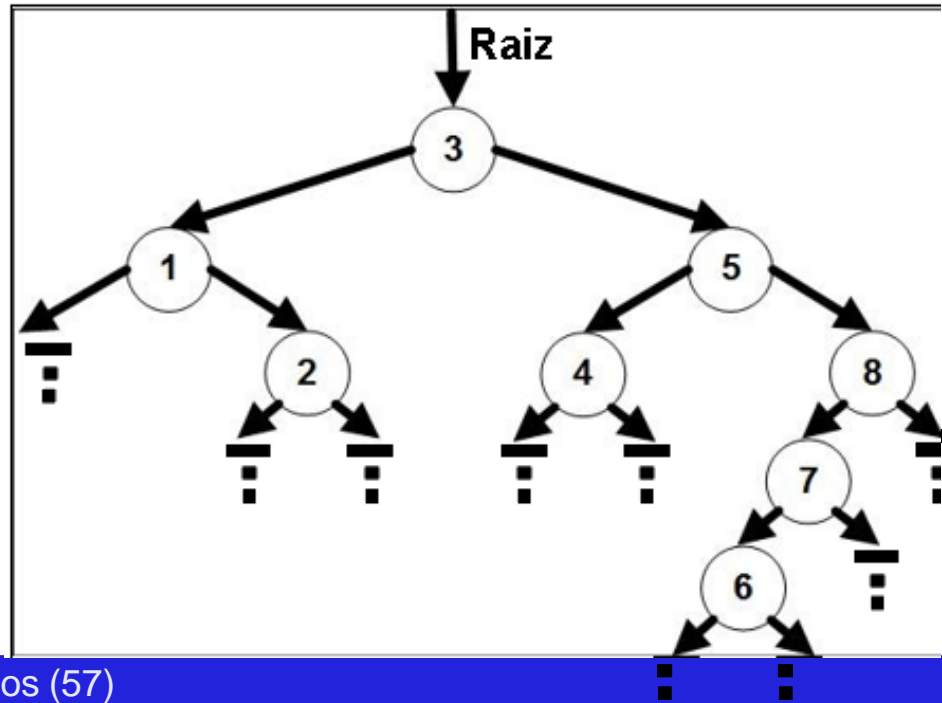


# Classe Árvore Binária: Remoção

```
class ArvoreBinaria
{
    private No raiz;
    public ArvoreBinaria(){raiz = null;}
    public void Inserir(int x) { }
    public bool Pesquisar(int x) { }
    public void CaminharCentral() { }
    public void CaminharPre() { }
    public void CaminharPos() { }
    public void Remover(int x) { }
}
```

Vamos Remover o 1  
(tem um filho) de nossa  
árvore

raiz

 $n(3)$ 

# Algoritmo de Remoção

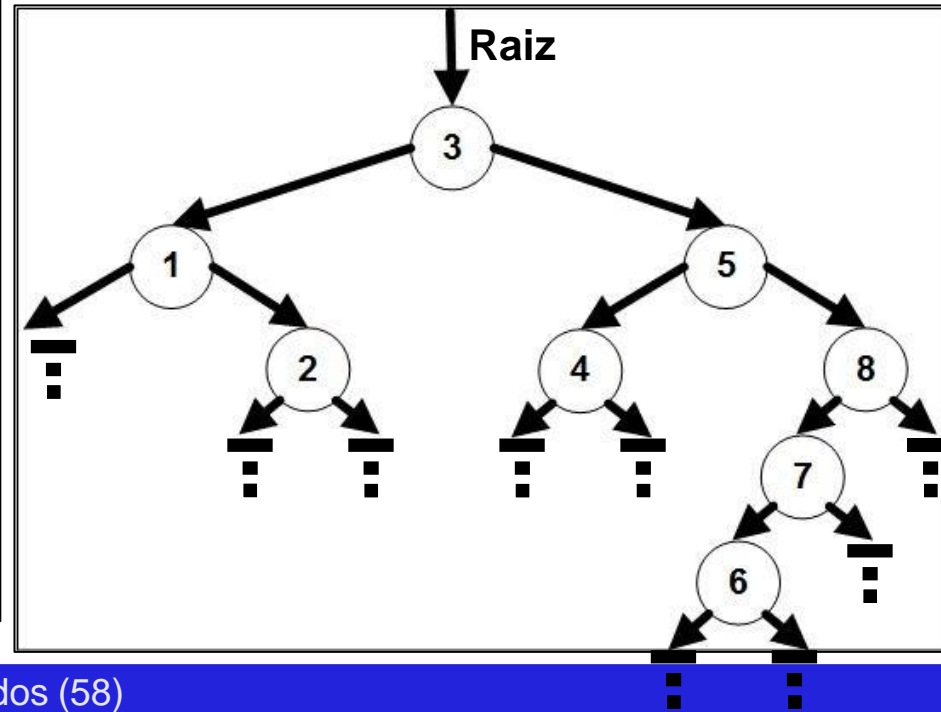
//Remover(1), um filho

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}

No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```

raiz n(3) x 1



# Algoritmo de Remoção

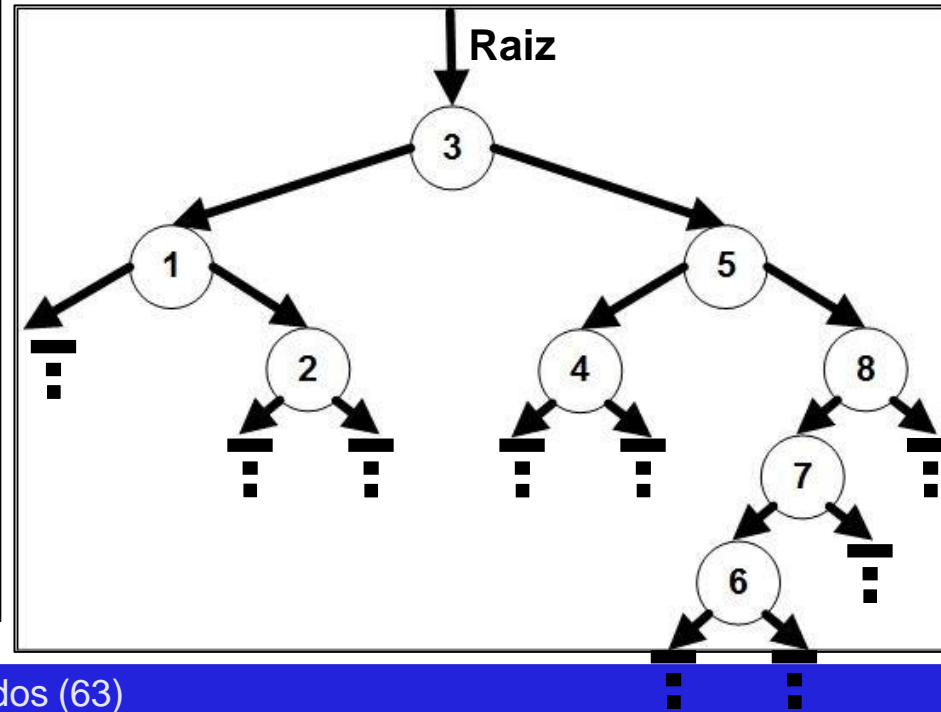
//Remover(1), um filho

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}

No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```

raiz    n(3)    x    1



# Algoritmo de Remoção

//Remover(1), um filho

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}
```

```
private No Remover(int x, No i) {
```

```
    if (i == null) {        throw new Exception("Erro!");
    } else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq);
    } else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir);
    } else if (i.Dir == null) {    i = i.Esq;
    } else if (i.Esq == null) {    i = i.Dir;
    } else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
```

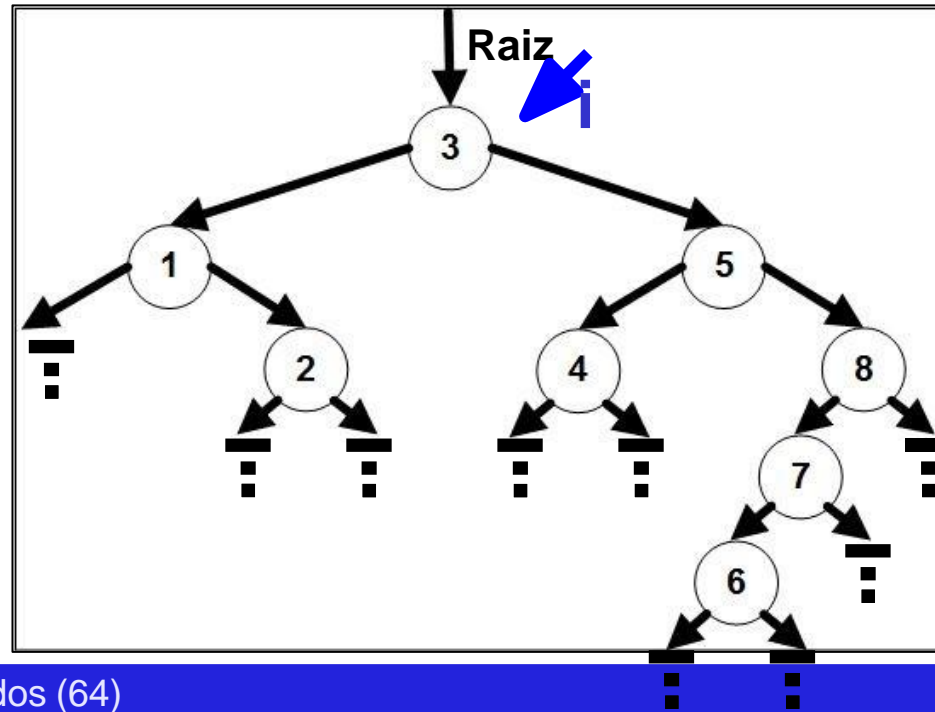
```
}
```

```
No MaiorEsq(No i, No j) {
```

```
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else {                j.Dir = MaiorEsq(i, j.Dir);        }
    return j;
```

```
}
```

raiz    n(3)    x    1    x    1    i    n(3)



# Algoritmo de Remoção

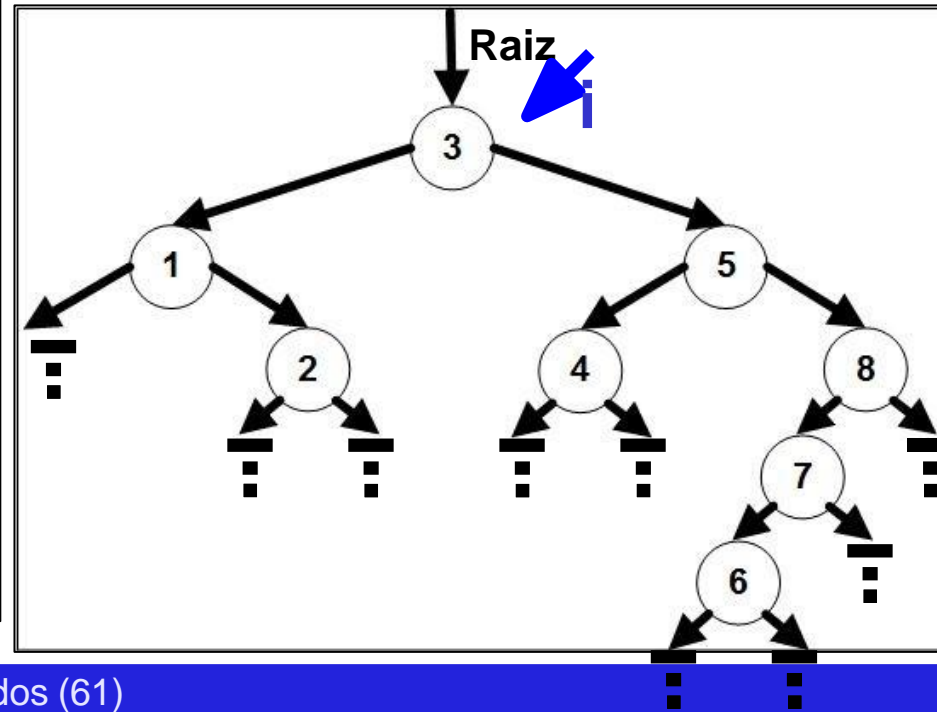
//Remover(1), um filho

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!");
    } else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq);
    } else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir);
    } else if (i.Dir == null) { i = i.Esq;
    } else if (i.Esq == null) { i = i.Dir;
    } else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}

No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```

raiz    n(3)    x    1    x    1    i    n(3)



# Algoritmo de Remoção

//Remover(1), um filho

```

public void Remover(int x) {
    raiz = Remover(x, raiz);
}

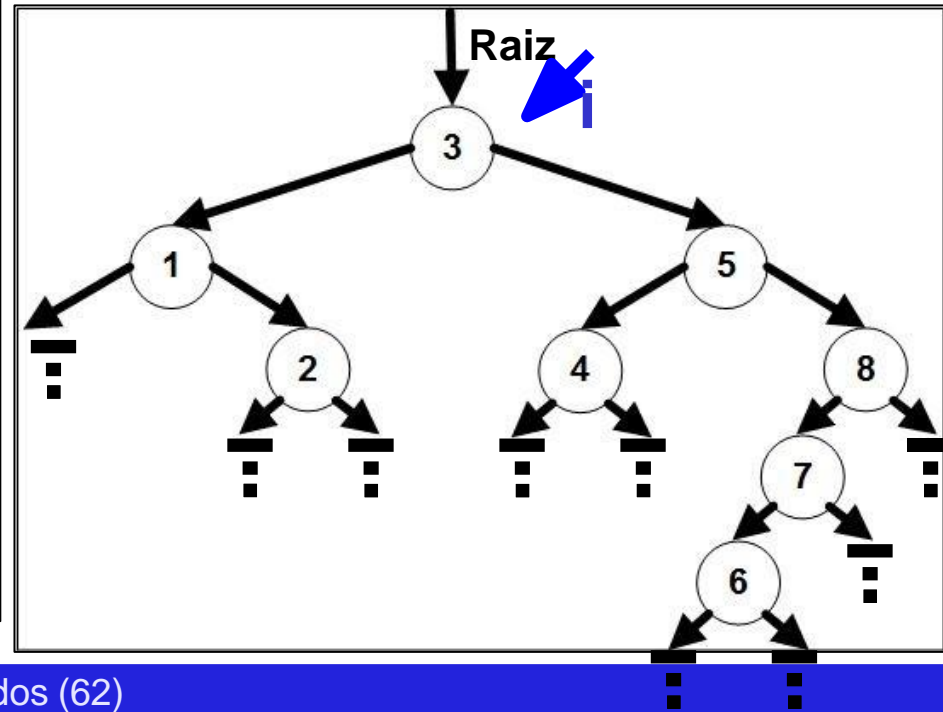
private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}

No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}

```

true:  $1 < 3$

raiz     $\boxed{n(3)}$      $\times$      $\boxed{1}$      $\times$      $\boxed{1}$      $i$      $\boxed{n(3)}$



# Algoritmo de Remoção

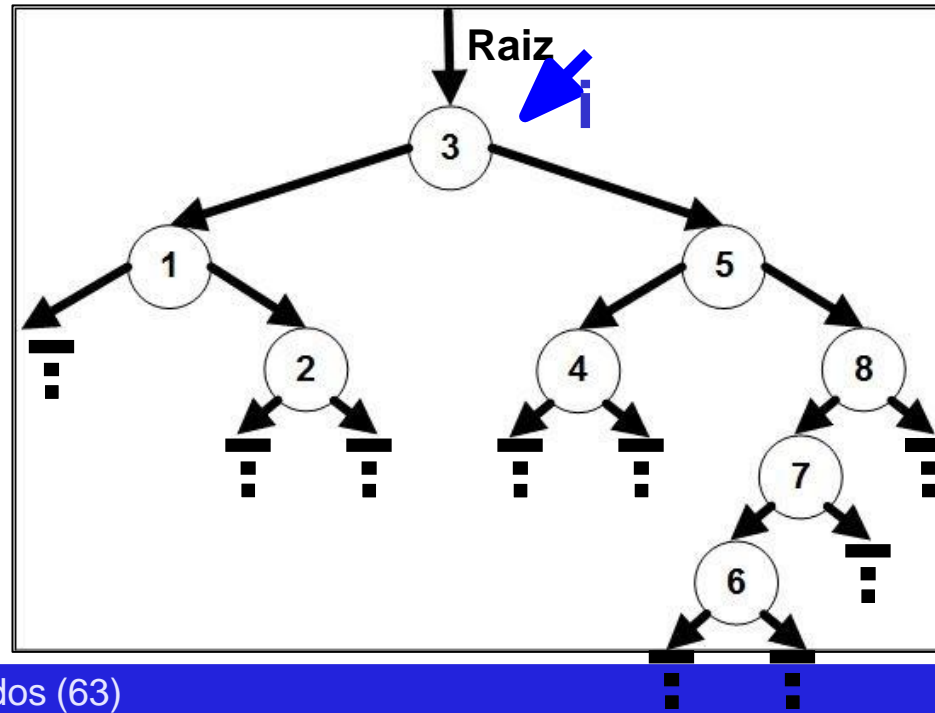
//Remover(1), um filho

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}

No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```

raiz    n(3)    x    1    x    1    i    n(3)



# Algoritmo de Remoção

//Remover(1), um filho

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}
```

```
private No Remover(int x, No i) {
```

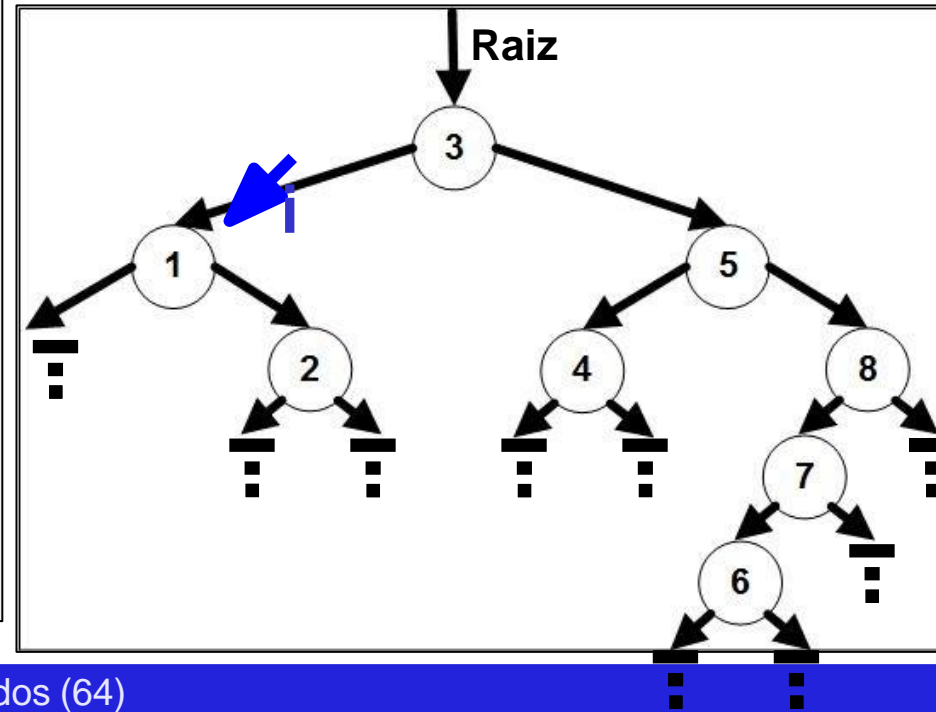
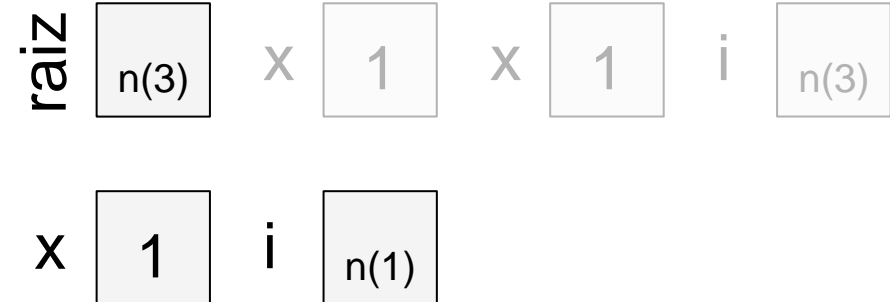
```
    if (i == null) {        throw new Exception("Erro!");
    } else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq);
    } else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir);
    } else if (i.Dir == null) {    i = i.Esq;
    } else if (i.Esq == null) {    i = i.Dir;
    } else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
```

```
}
```

```
No MaiorEsq(No i, No j) {
```

```
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else {                j.Dir = MaiorEsq(i, j.Dir);        }
    return j;
```

```
}
```





# Algoritmo de Remoção

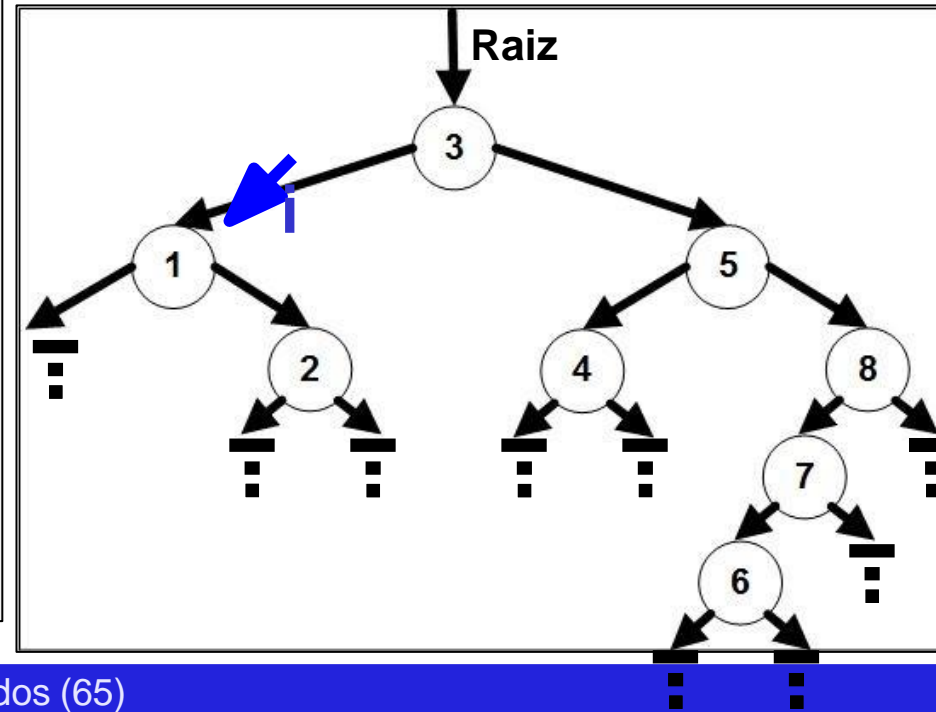
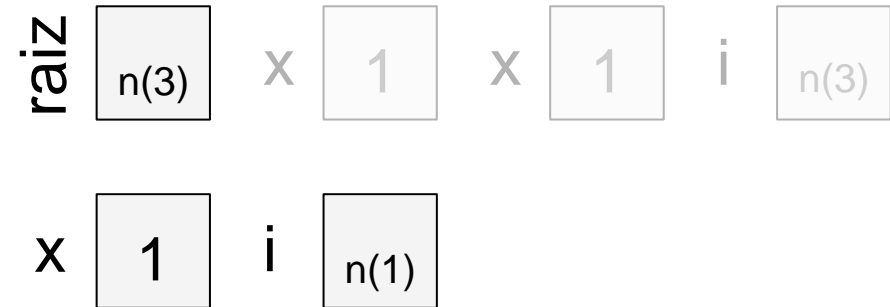
//Remover(1), um filho

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}

No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```

false: n(1) == null



# Algoritmo de Remoção

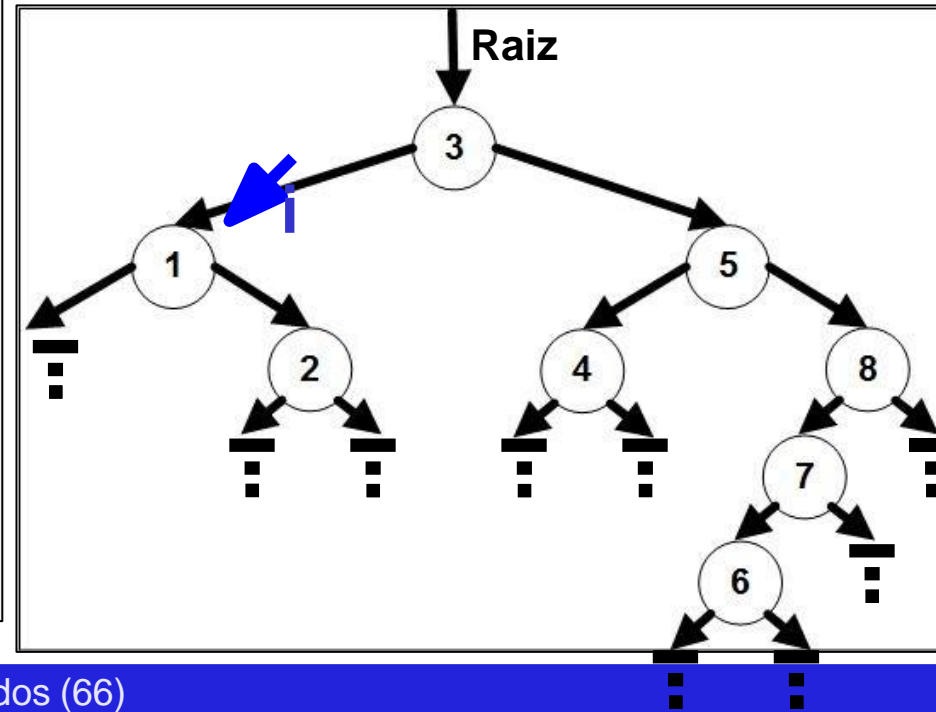
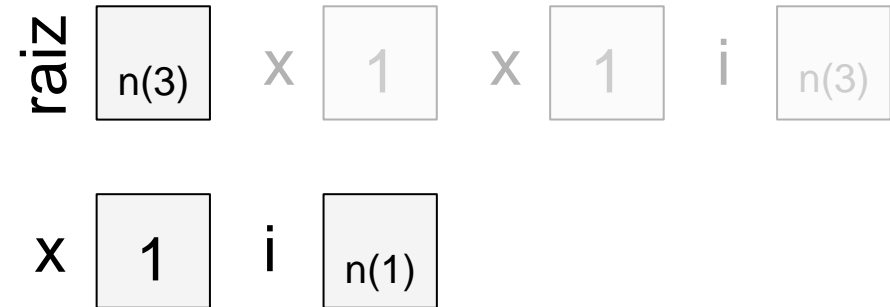
//Remover(1), um filho

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}

No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```

false: 1 < 1



# Algoritmo de Remoção

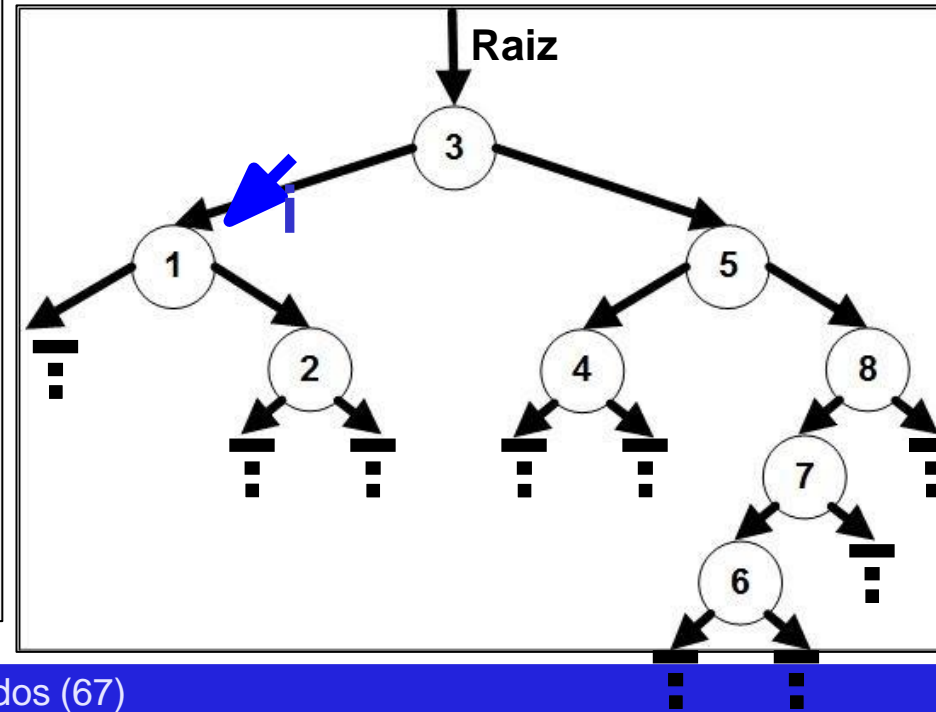
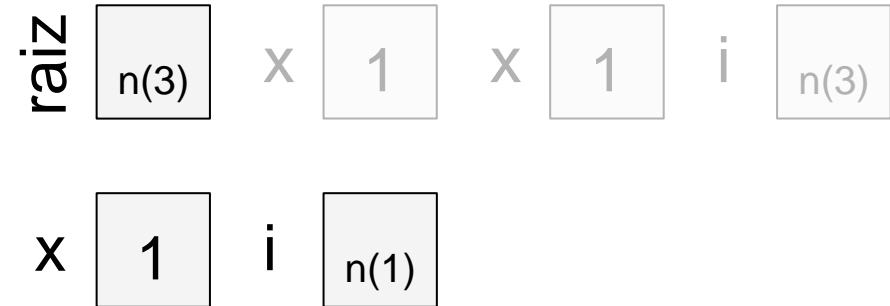
//Remover(1), um filho

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}

No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```

false: 1 > 1



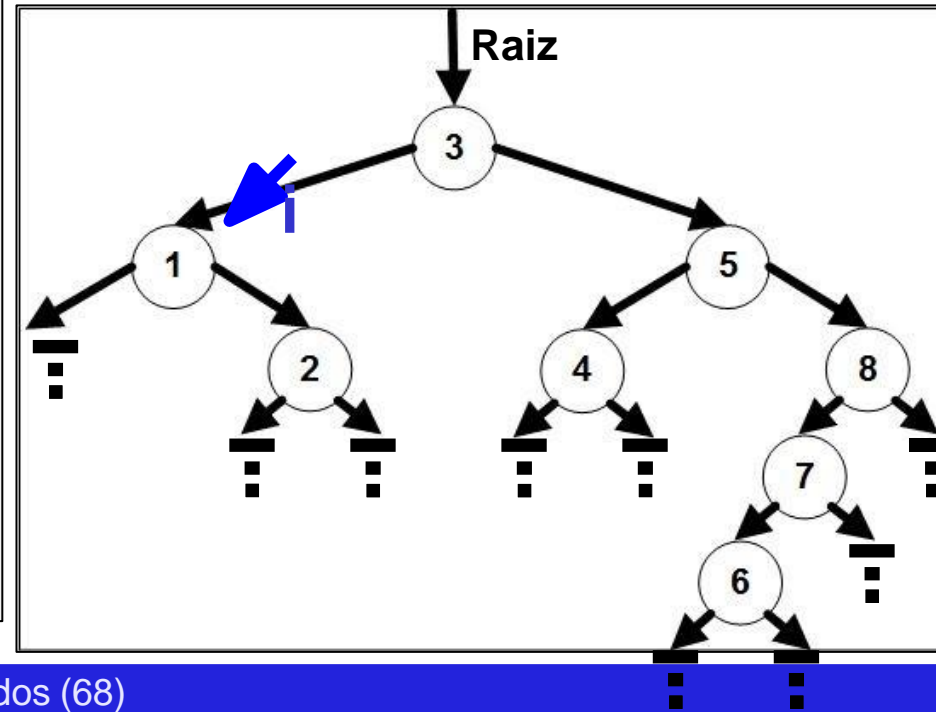
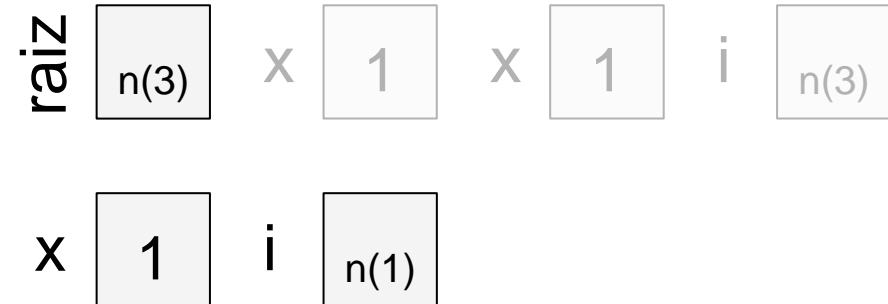
# Algoritmo de Remoção

//Remover(1), um filho

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}

No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```



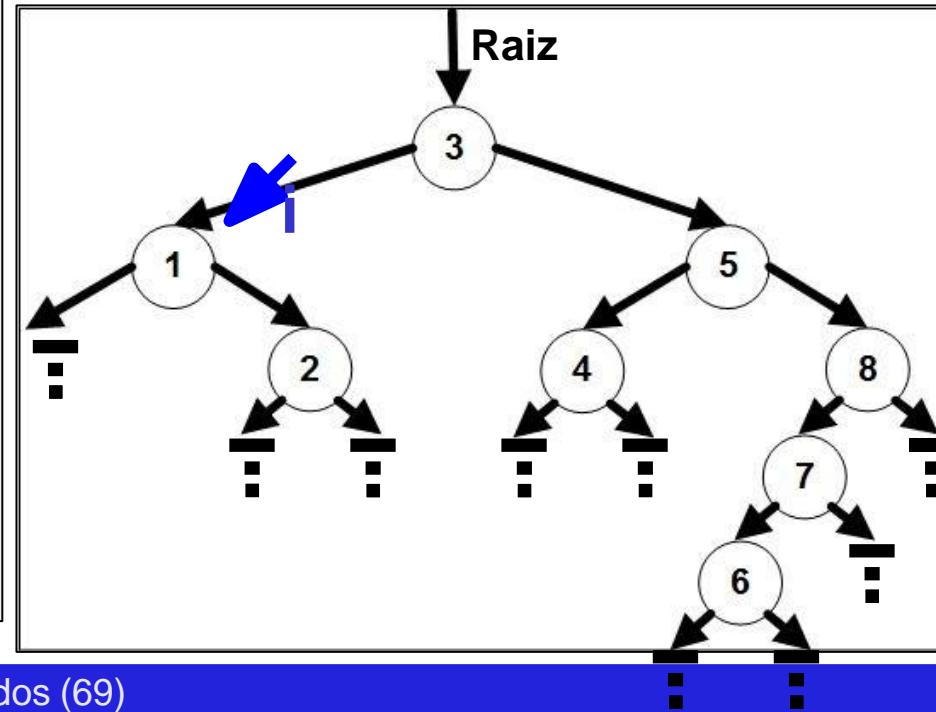
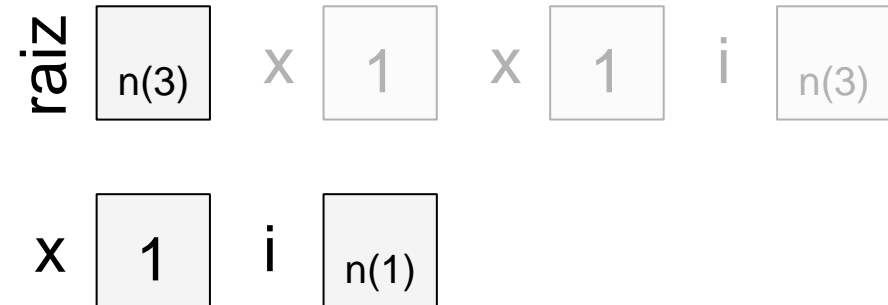
# Algoritmo de Remoção

//Remover(1), um filho

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) i = i.Esq;
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}

No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```



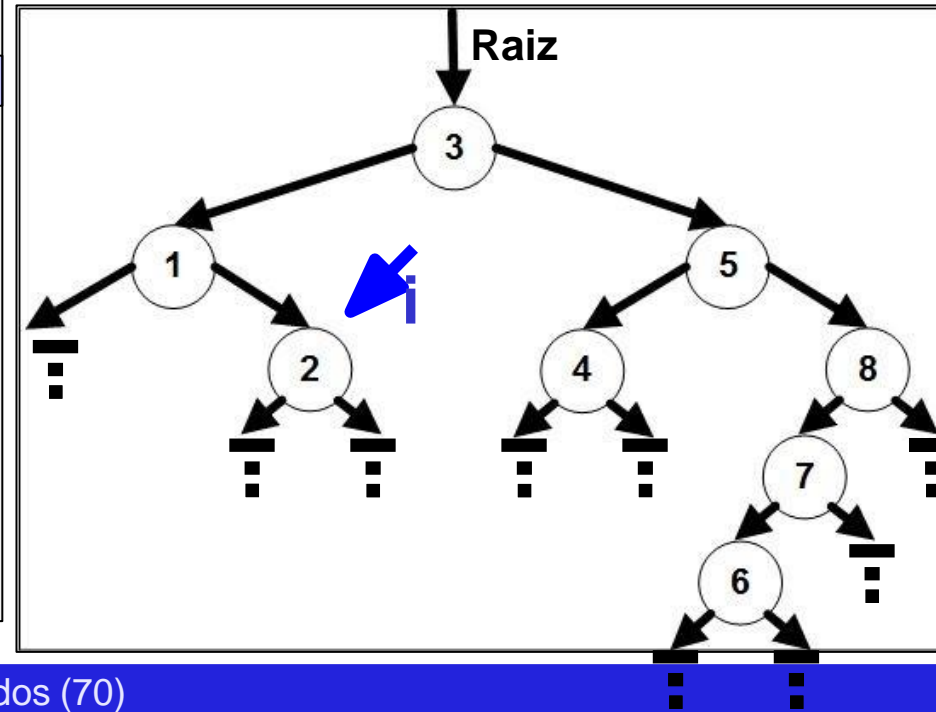
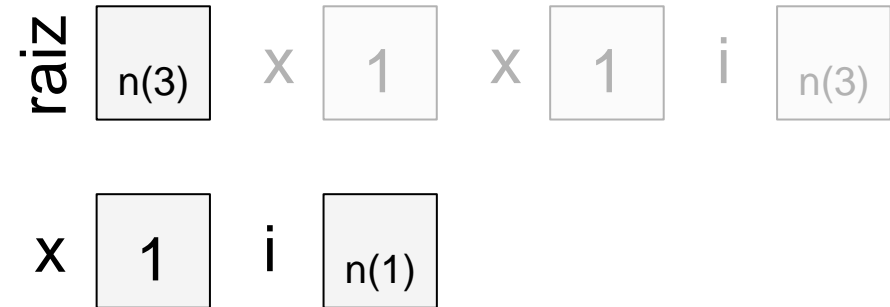
# Algoritmo de Remoção

//Remover(1), um filho

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}

No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```



# Algoritmo de Remoção

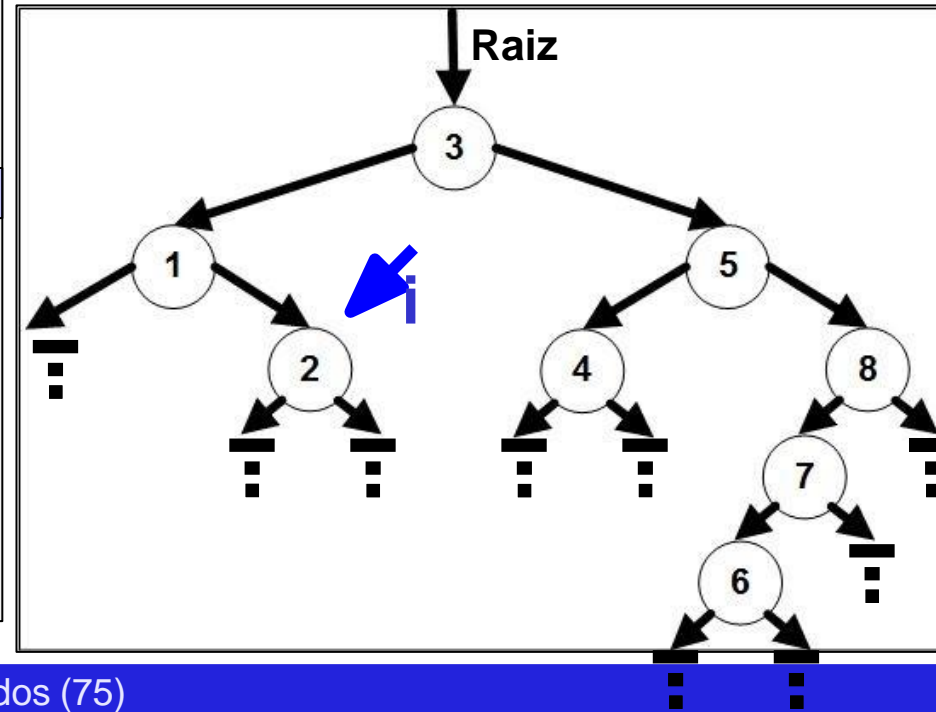
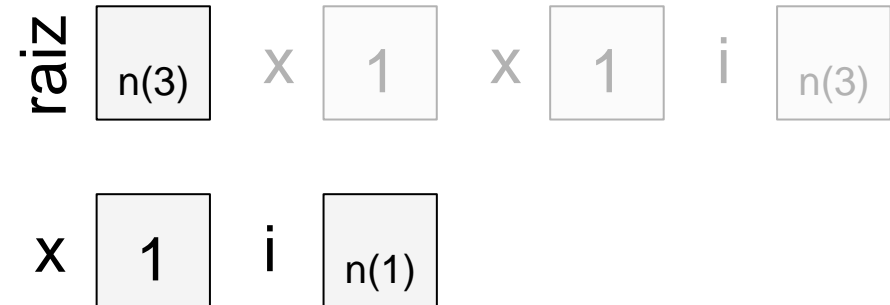
//Remover(1), um filho

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}
```

Retorna n(2)

```
No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```



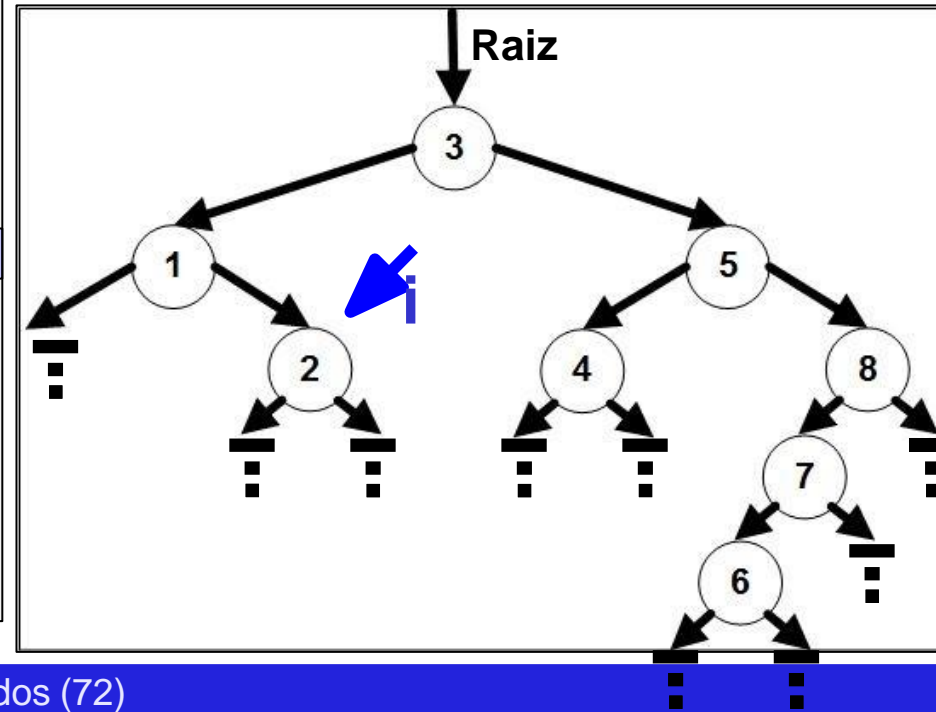
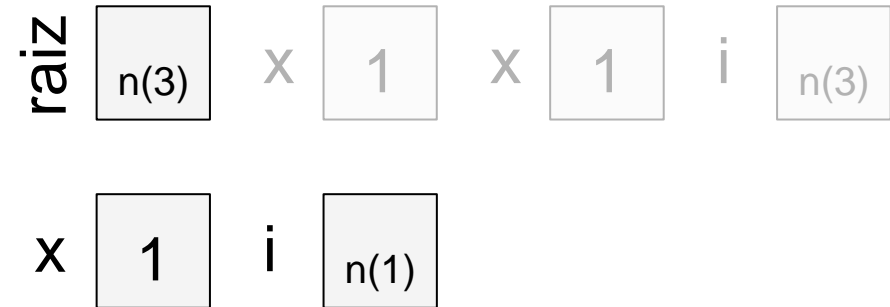
# Algoritmo de Remoção

//Remover(1), um filho

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}
```

```
No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```





# Algoritmo de Remoção

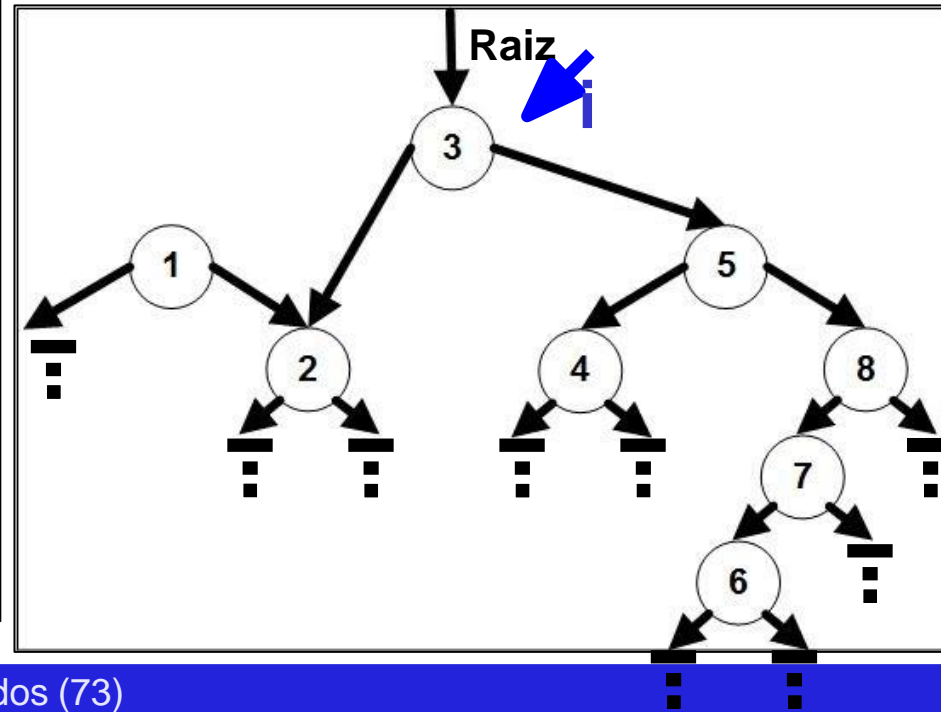
//Remover(1), um filho

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}

No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```

raiz    n(3)    x    1    x    1    i    n(3)



# Algoritmo de Remoção

//Remover(1), um filho

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}
```

```
private int
```

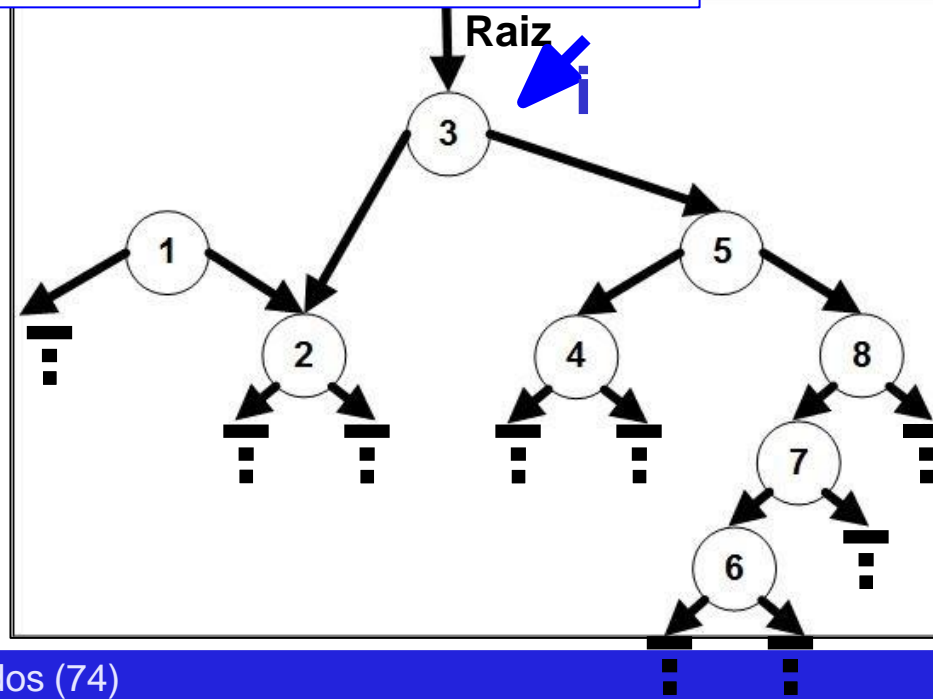
```
if (i == null) {
} else if (i.Esq == null) {
} else if (i.Dir == null) {
} else if (i.Esq == null) {
} else {
    i.Esq = MaiorEsq(i, i.Esq);
}
return i;
```

```
No MaiorEsq(No i, No j) {
```

```
if (j.Dir == null) { i.Elemento=j.Elemento; j=j.Esq; }
else {
    j.Dir = MaiorEsq(i, j.Dir);
}
return j;
}
```

raiz    n(3)    x    1    x    1    i    n(3)

Após a coleta de lixo  
(que não controlamos quando ela acontece)...



# Algoritmo de Remoção

//Remover(1), um filho

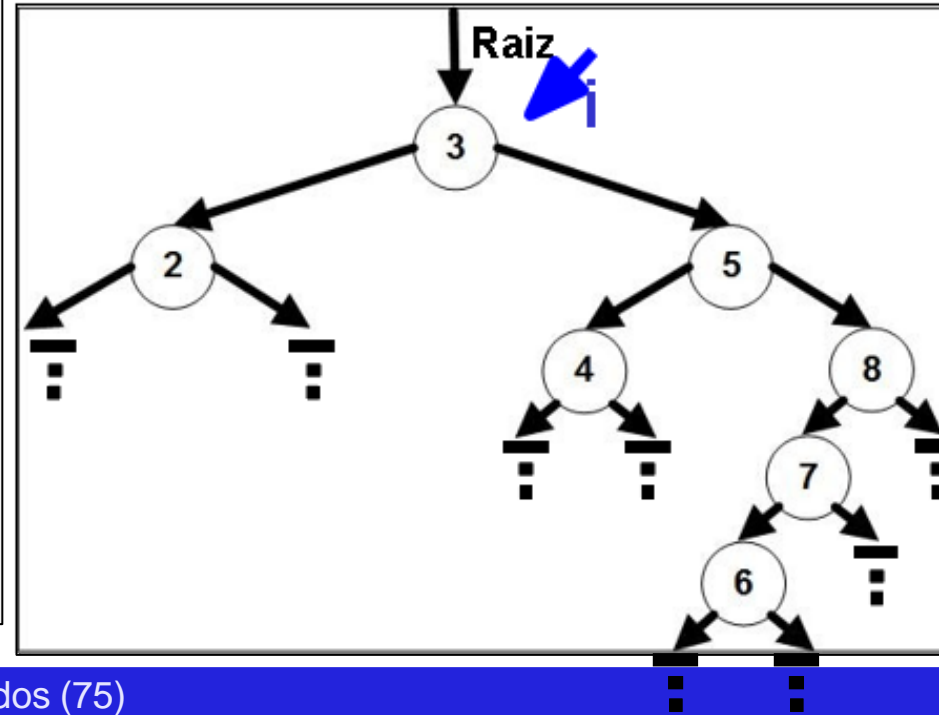
```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}
```

```
private int Remover(int x, int i) {
    if (i == null) return null;
    if (i.Esq == null) {
        if (i.Dir == null) return i;
        else if (i.Dir.Esq == null) {
            i.Dir = i.Dir.Dir;
            return i;
        }
        else if (i.Dir.Dir.Esq == null) {
            i.Dir.Dir = i.Dir.Dir.Dir;
            return i;
        }
        else {
            i.Dir = i.Dir.Dir;
            return i;
        }
    }
    else if (i.Dir == null) {
        i = i.Esq;
    }
    else if (i.Esq == null) {
        i = i.Dir;
    }
    else {
        i.Esq = MaiorEsq(i, i.Esq);
    }
    return i;
}
```

```
No MaiorEsq(int i, int j) {
    if (j.Dir == null) {
        i.Elemento = j.Elemento;
        j = j.Esq;
    }
    else {
        j.Dir = MaiorEsq(i, j.Dir);
    }
    return j;
}
```

raiz    n(3)    x    1    x    1    i    n(3)

De uma forma mais organizada ...



# Algoritmo de Remoção

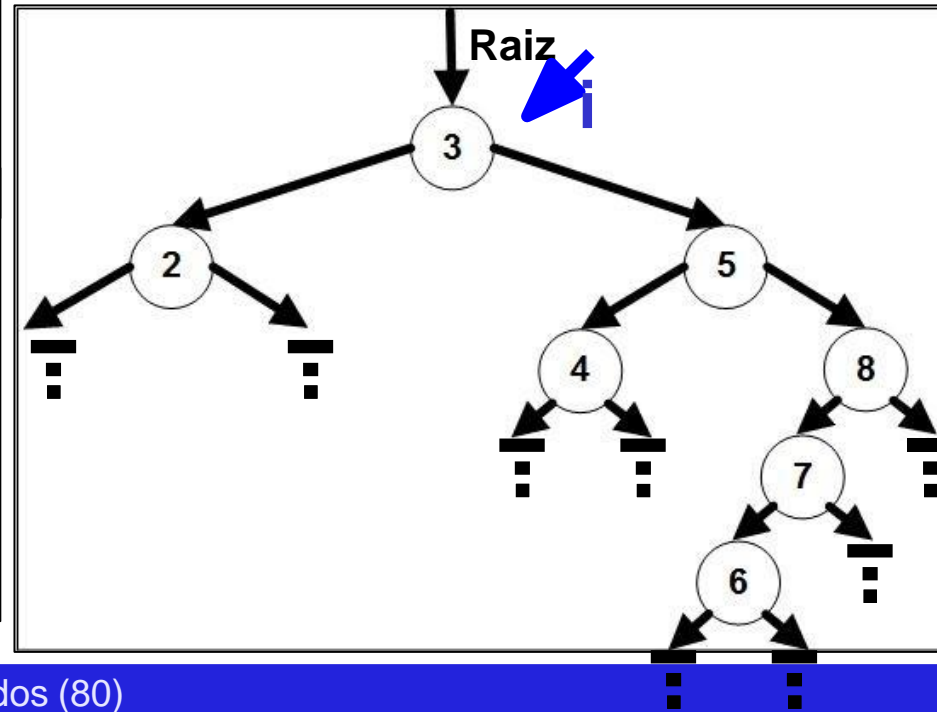
//Remover(1), um filho

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}
```

Retorna n(3)

```
No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```



# Algoritmo de Remoção

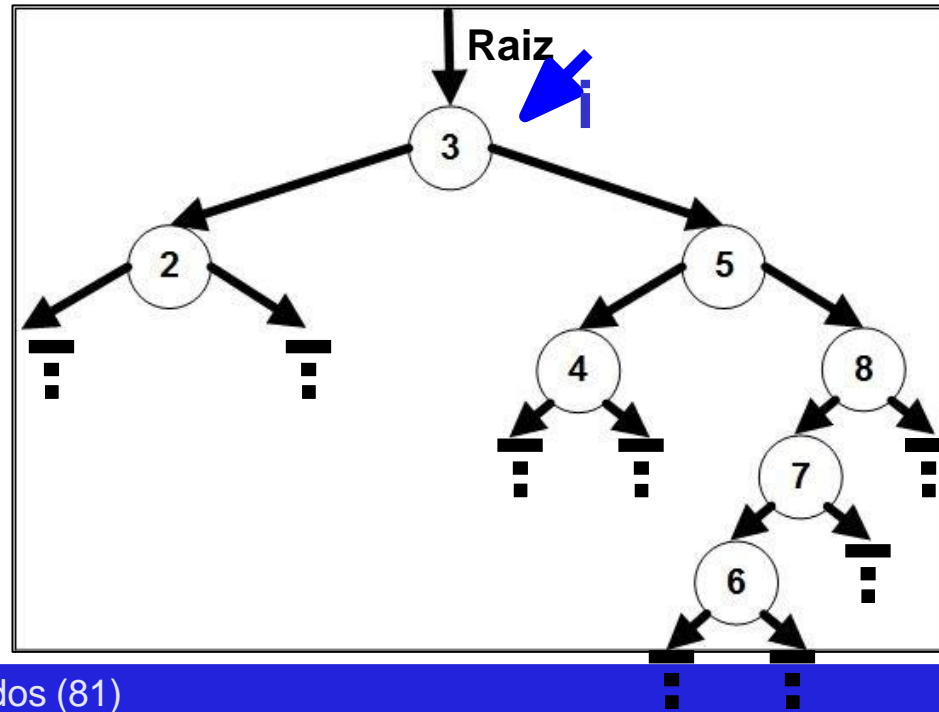
//Remover(1), um filho

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}
```

```
No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```

raiz    n(3)    x    1    x    1    i    n(3)



# Algoritmo de Remoção

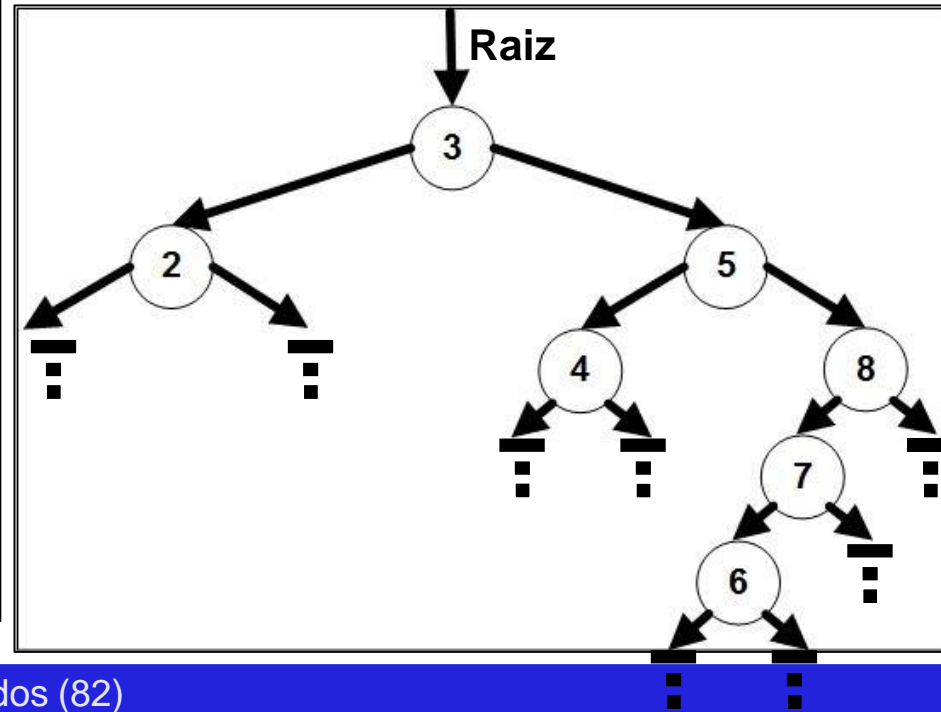
//Remover(1), um filho

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}

No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```

raiz    n(3)    x    1



# Algoritmo de Remoção

//Remover(1), um filho

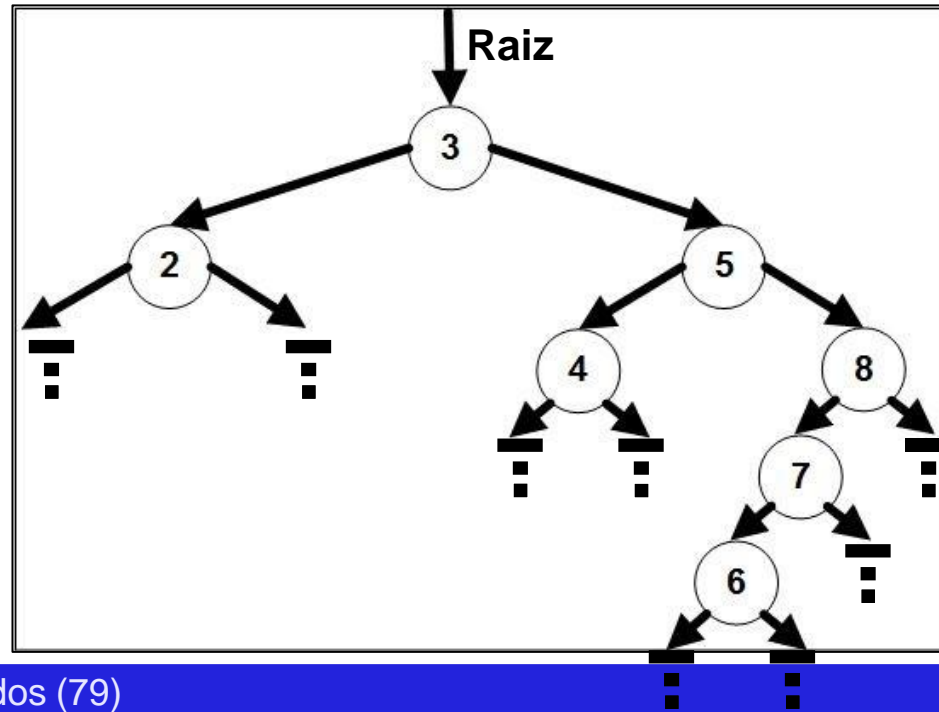
```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}
```

```
private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}
```

```
No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```

raiz

n(3)



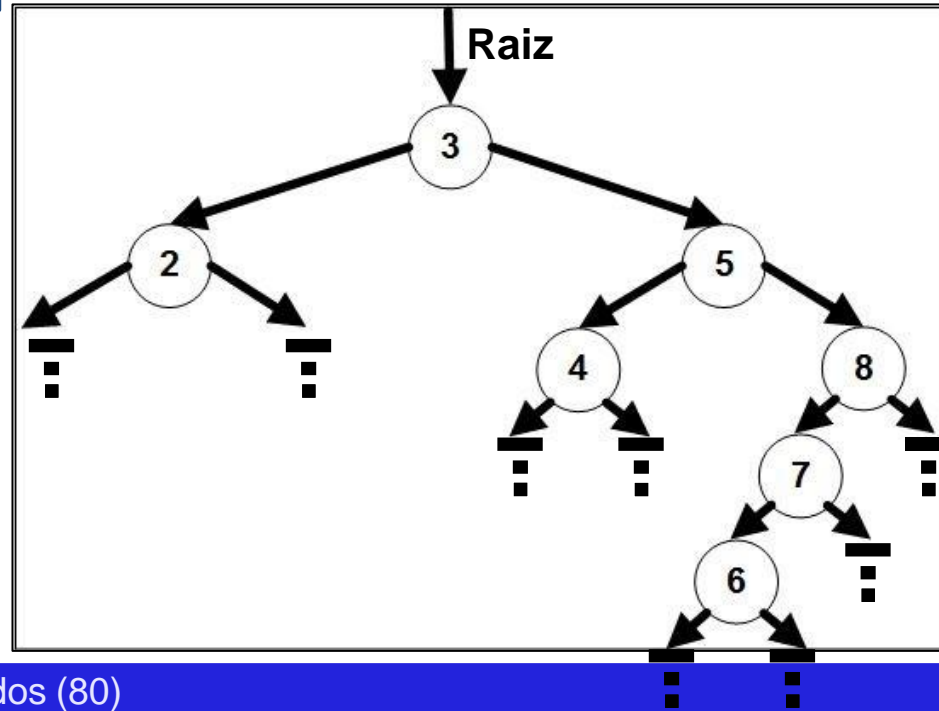
# Algoritmo de Remoção

```
class ArvoreBinaria
{
    private No raiz;
    public ArvoreBinaria(){raiz = null;}
    public void Inserir(int x) { }
    public bool Pesquisar(int x) { }
    public void CaminharCentral() { }
    public void CaminharPre() { }
    public void CaminharPos() { }
    public void Remover(int x) { }
```

Voltando com o 1 antes  
de fazer outra remoção

raiz

n(3)

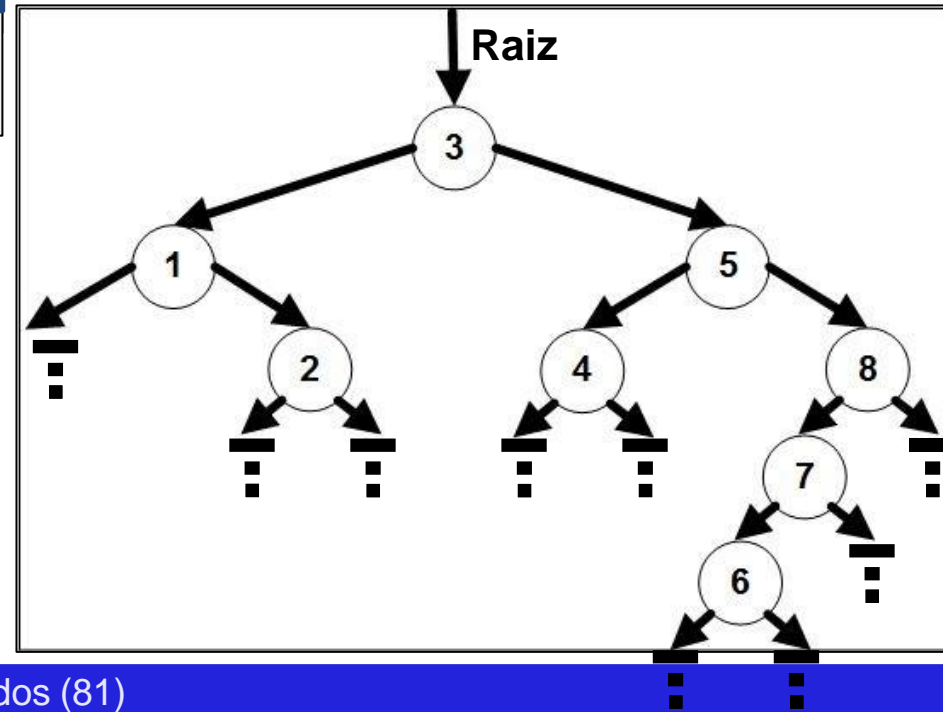
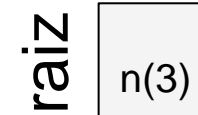




# Algoritmo de Remoção

```
class ArvoreBinaria
{
    private No raiz;
    public ArvoreBinaria(){raiz = null;}
    public void Inserir(int x) { }
    public bool Pesquisar(int x) { }
    public void CaminharCentral() { }
    public void CaminharPre() { }
    public void CaminharPos() { }
    public void Remover(int x) { }
}
```

Vamos Remover o 3 (tem dois filhos) de nossa árvore



# Algoritmo de Remoção

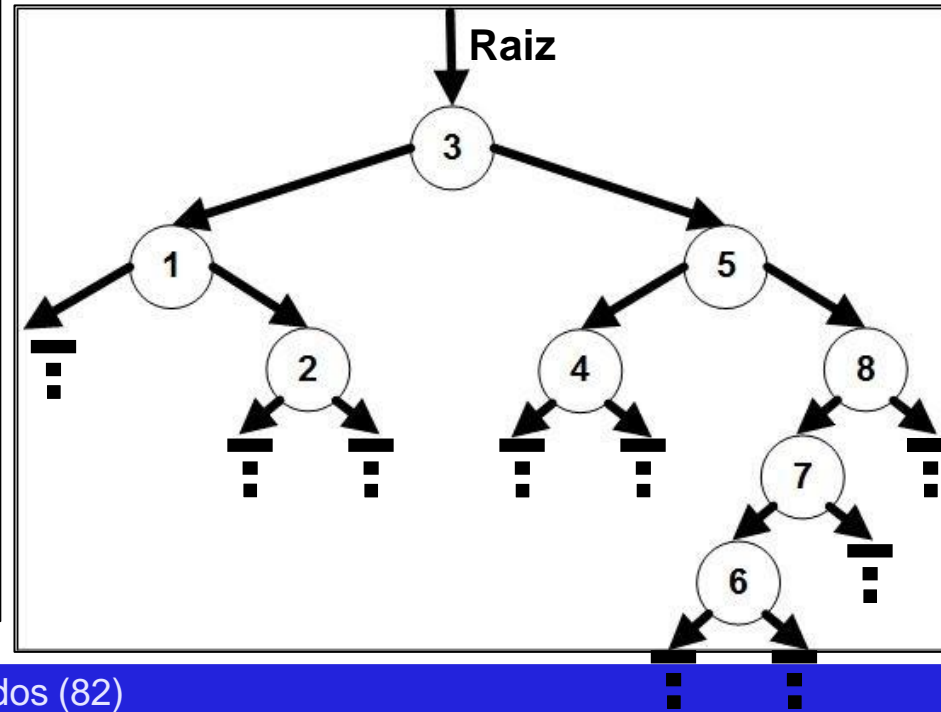
//Remover(3), dois filhos

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}

No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```

raiz n(3) x 3



# Algoritmo de Remoção

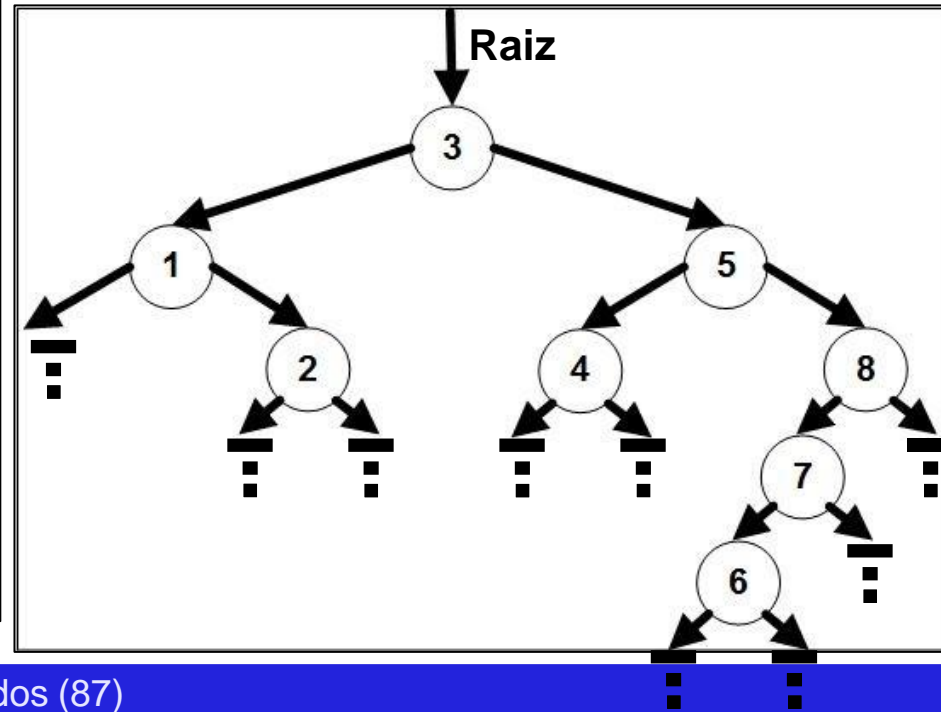
//Remover(3), dois filhos

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}

No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```

raiz     $n(3)$     x    3



# Algoritmo de Remoção

//Remover(3), dois filhos

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}
```

```
private No Remover(int x, No i) {
```

```
    if (i == null) {        throw new Exception("Erro!");
    } else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq);
    } else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir);
    } else if (i.Dir == null) {    i = i.Esq;
    } else if (i.Esq == null) {    i = i.Dir;
    } else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
```

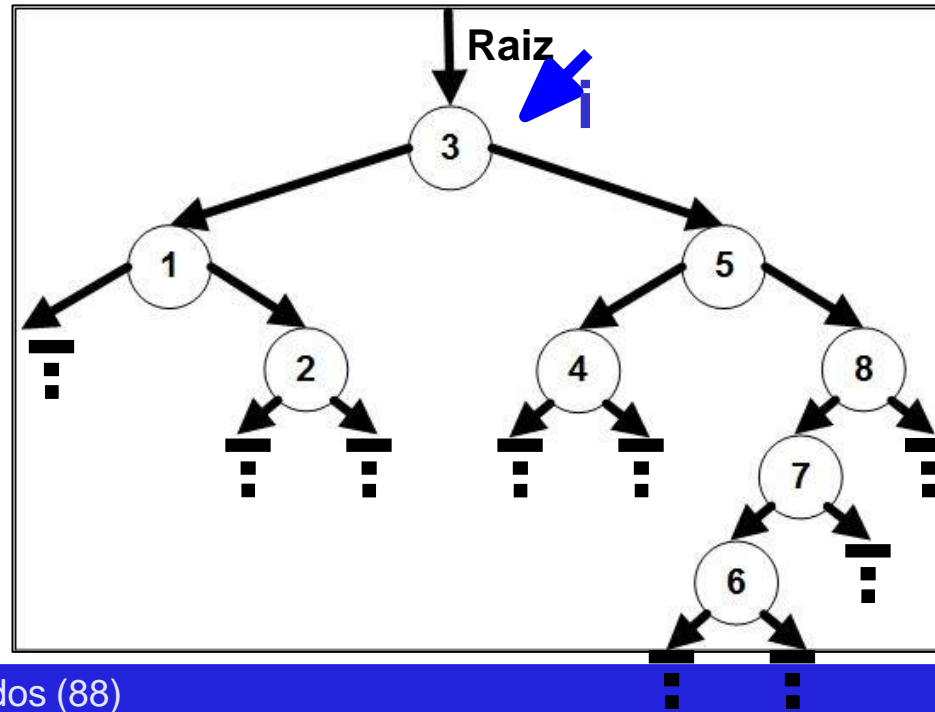
```
}
```

```
No MaiorEsq(No i, No j) {
```

```
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else {                j.Dir = MaiorEsq(i, j.Dir);        }
    return j;
```

```
}
```

raiz    n(3)    x    3    x    3    i    n(3)



# Algoritmo de Remoção

//Remover(3), dois filhos

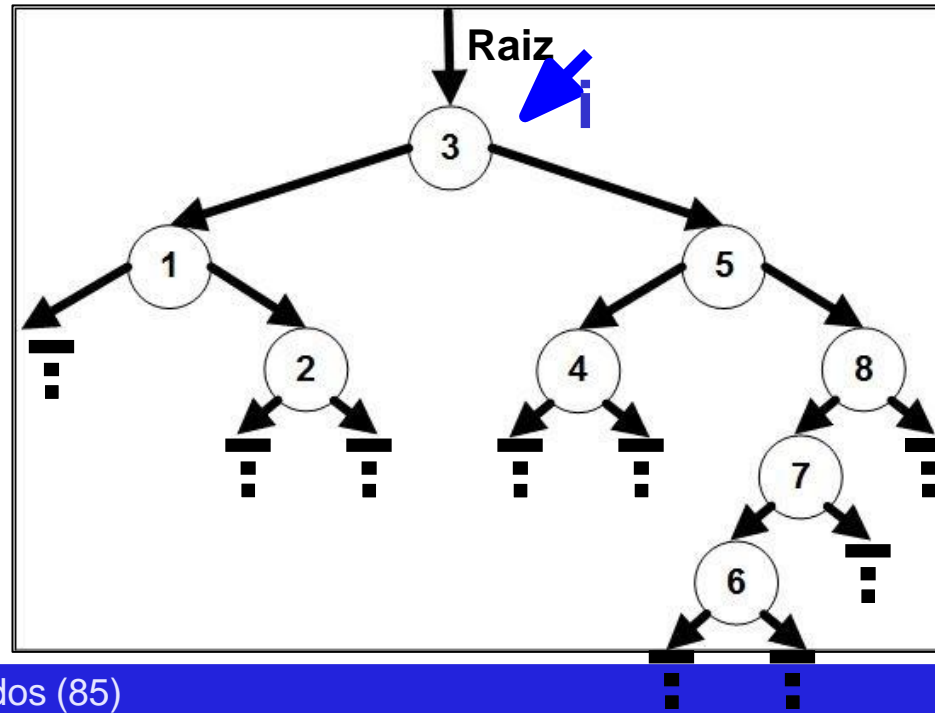
```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!");
    } else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq);
    } else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir);
    } else if (i.Dir == null) { i = i.Esq;
    } else if (i.Esq == null) { i = i.Dir;
    } else { i.Esq = MaiorEsq(i, i.Esq);
    }
    return i;
}

No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq;
    } else {
        j.Dir = MaiorEsq(i, j.Dir);
    }
    return j;
}
```

false:  $n(3) == \text{null}$

raiz  $n(3)$  x  $3$  x  $3$  i  $n(3)$



# Algoritmo de Remoção

//Remover(3), dois filhos

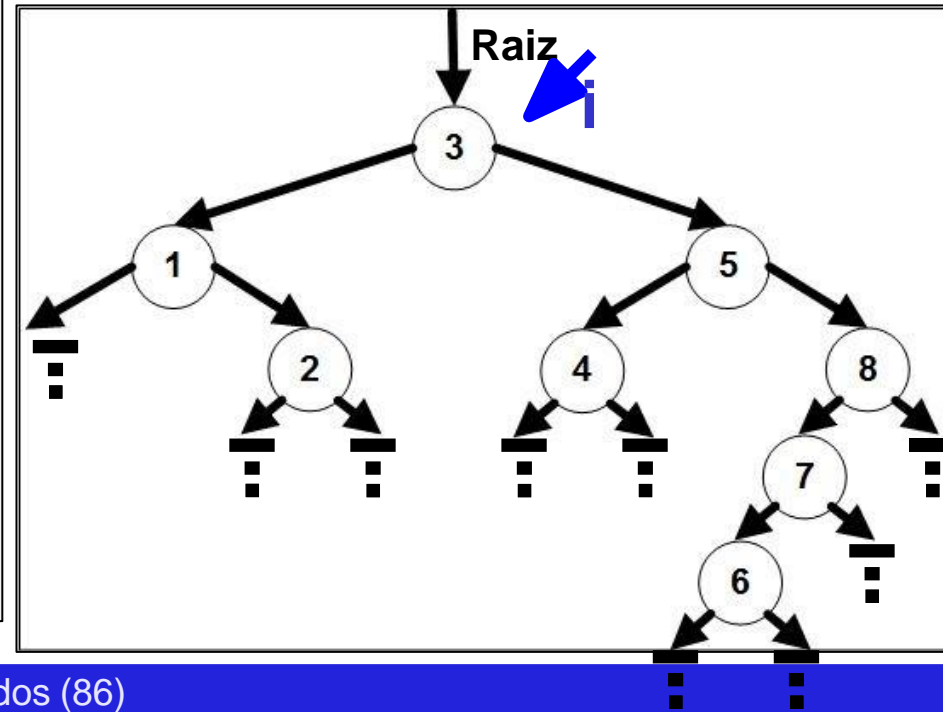
```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}

No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```

false: 3 < 3

raiz    n(3)    x    3    x    3    i    n(3)



# Algoritmo de Remoção

//Remover(3), dois filhos

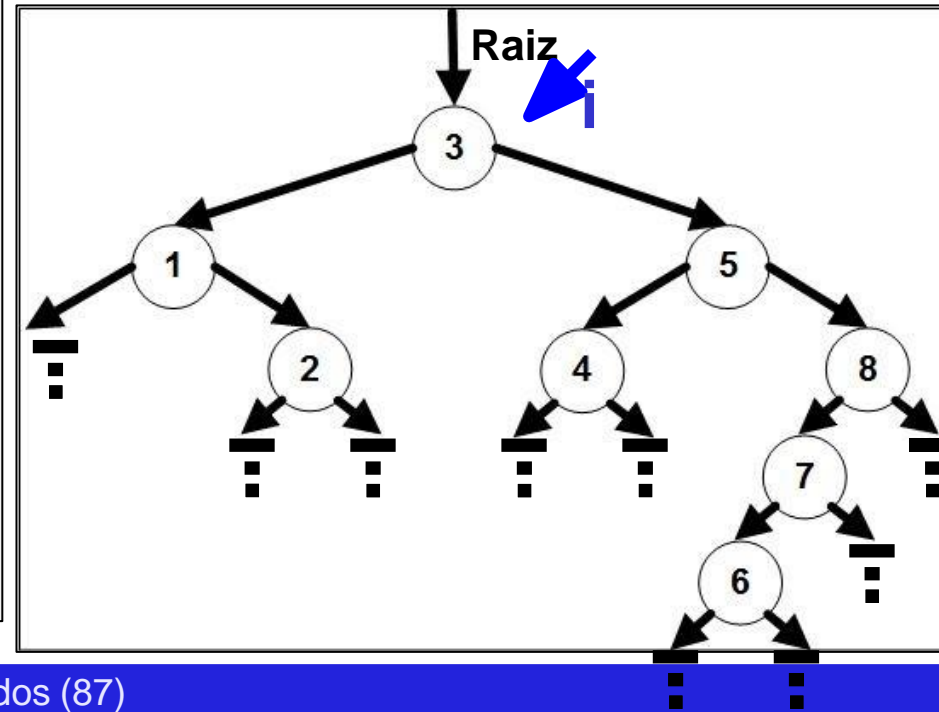
```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}

No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```

false: 3 > 3

raiz    n(3)    x    3    x    3    i    n(3)



# Algoritmo de Remoção

//Remover(3), dois filhos

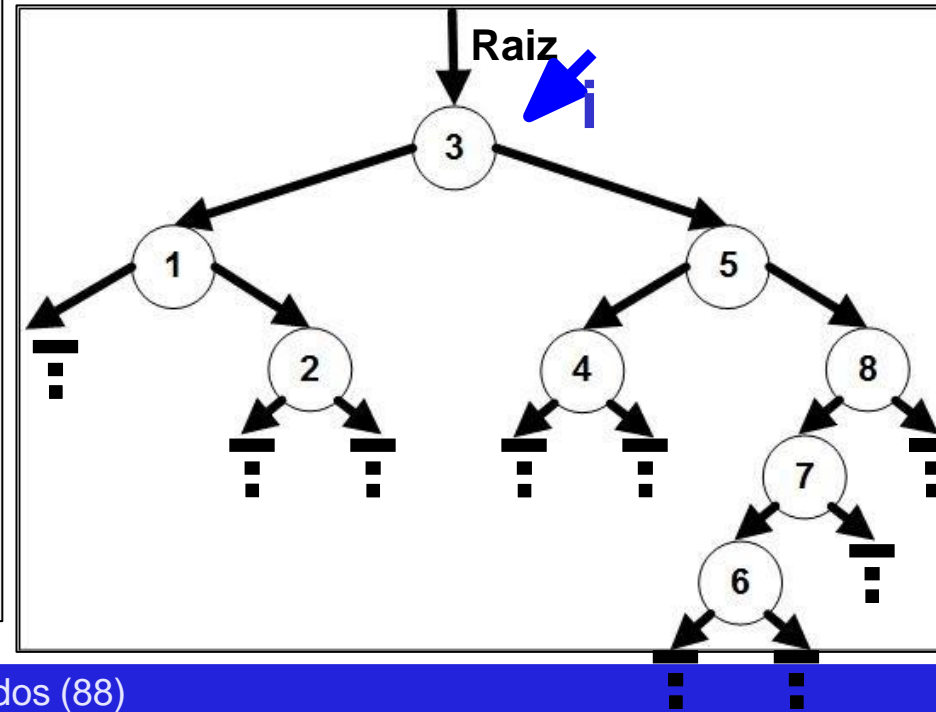
```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}

No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```

false: n(5) == false

raiz    n(3)    x    3    x    3    i    n(3)





# Algoritmo de Remoção

//Remover(3), dois filhos

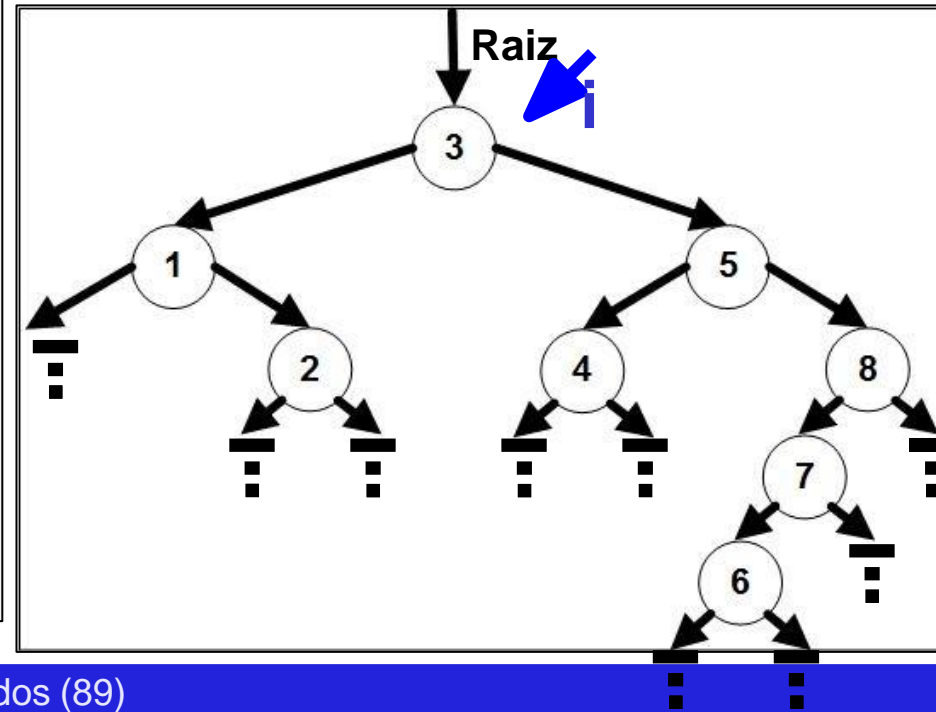
```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) i = i.Esq;
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}

No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```

false: n(1) == false

raiz    n(3)    x    3    x    3    i    n(3)



# Algoritmo de Remoção

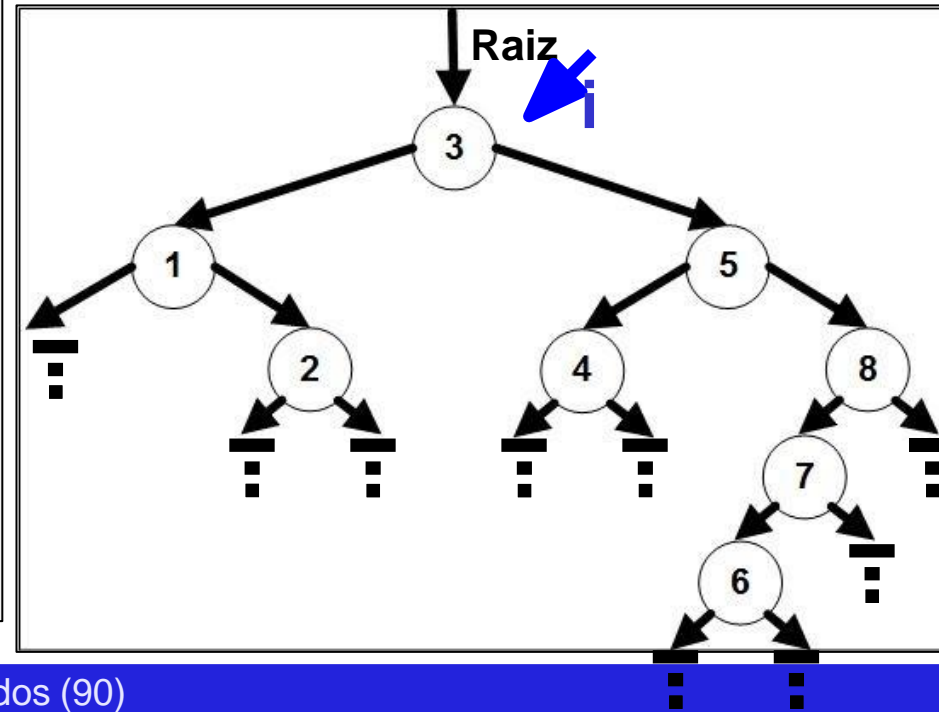
//Remover(3), dois filhos

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}

No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```

raiz    n(3)    x    3    x    3    i    n(3)



# Algoritmo de Remoção

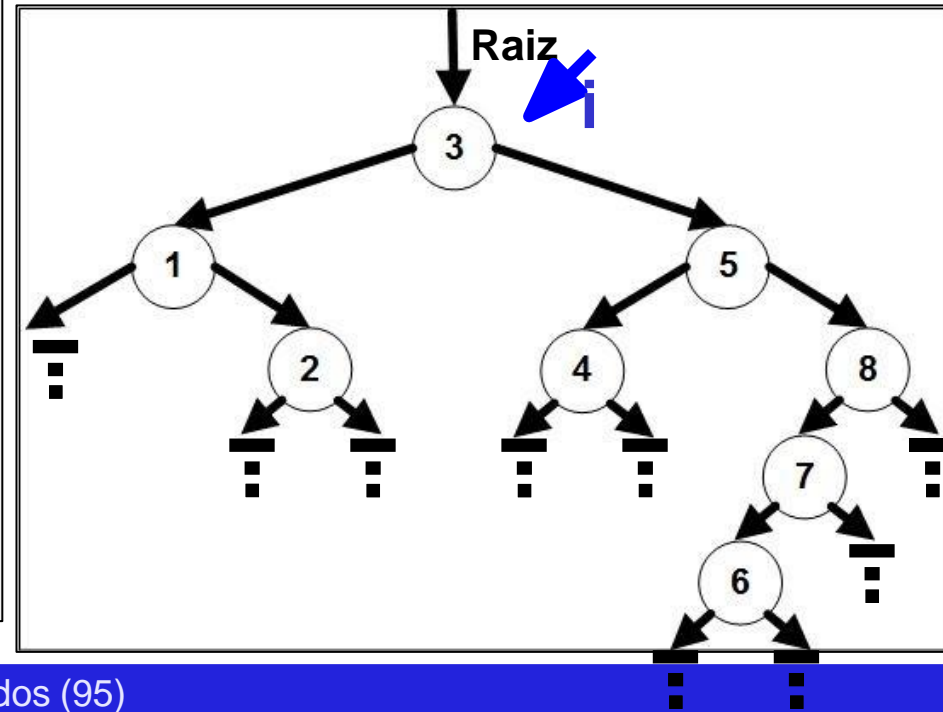
//Remover(3), dois filhos

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}

No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```

raiz    n(3)    x    3    x    3    i    n(3)



# Algoritmo de Remoção

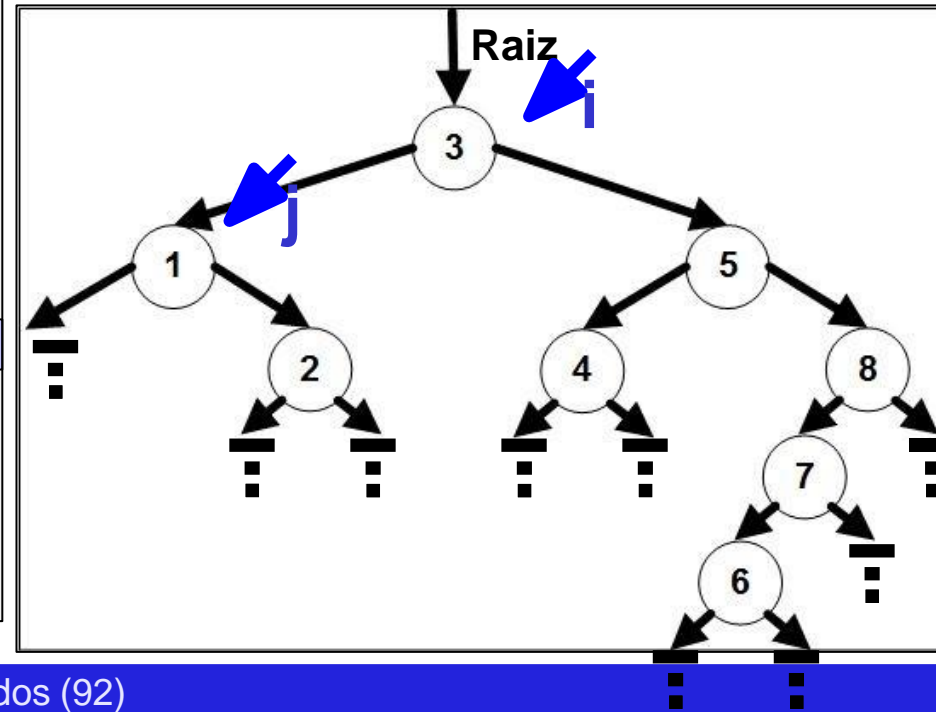
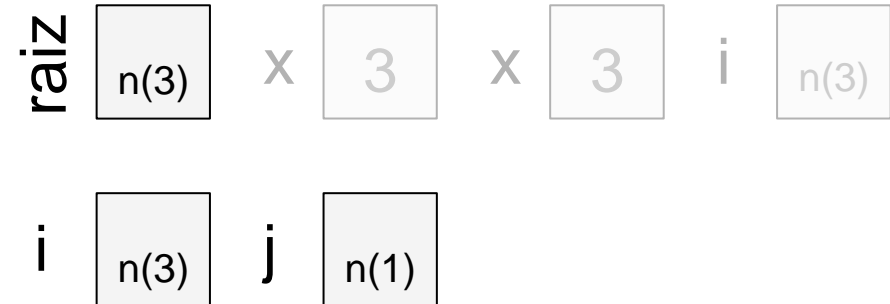
//Remover(3), dois filhos

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}
```

No MaiorEsq(No i, No j) {

```
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```



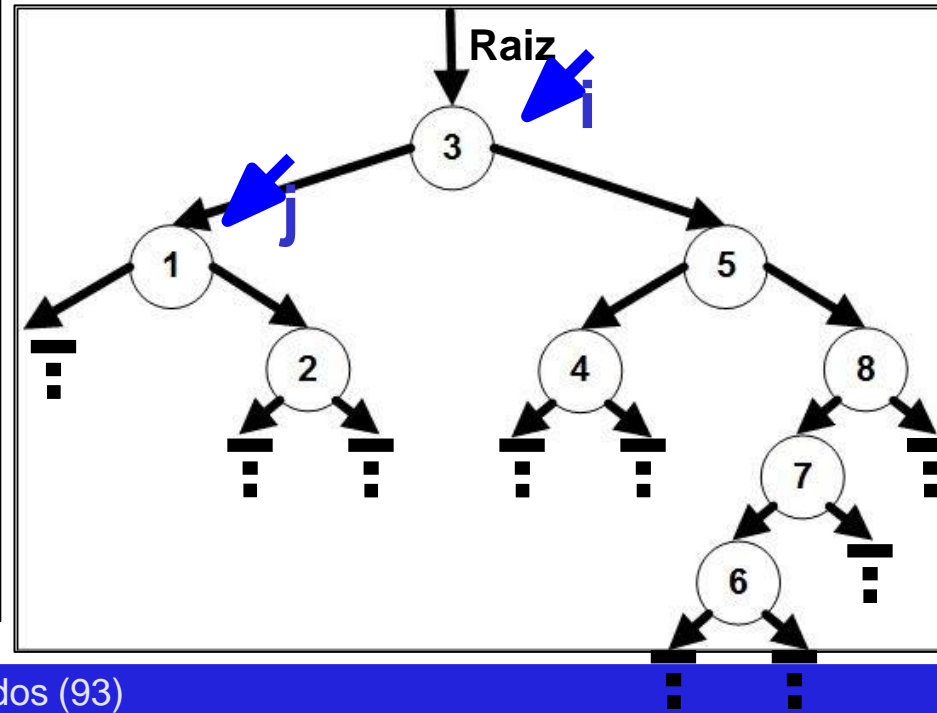
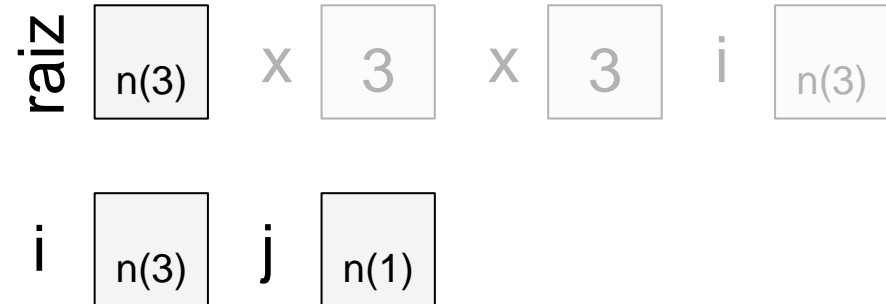
# Algoritmo de Remoção

//Remover(3), dois filhos

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}

No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```



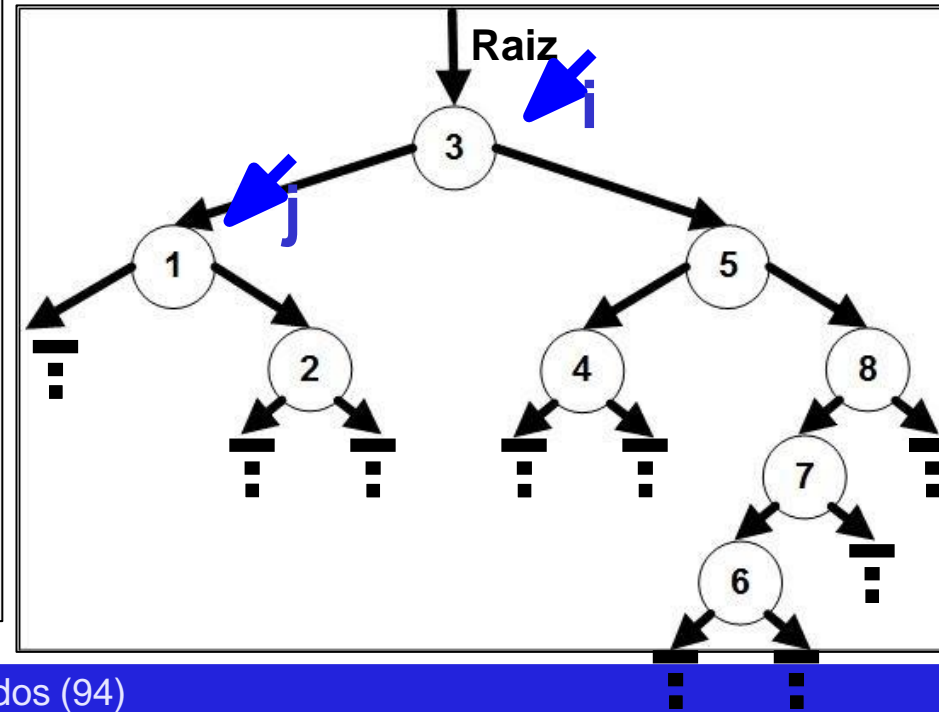
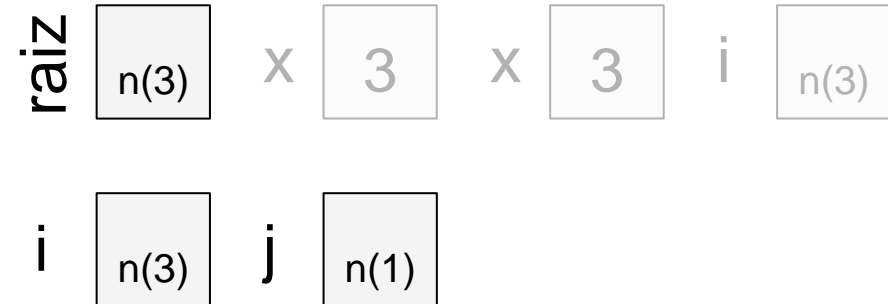
# Algoritmo de Remoção

//Remover(3), dois filhos

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}

No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```



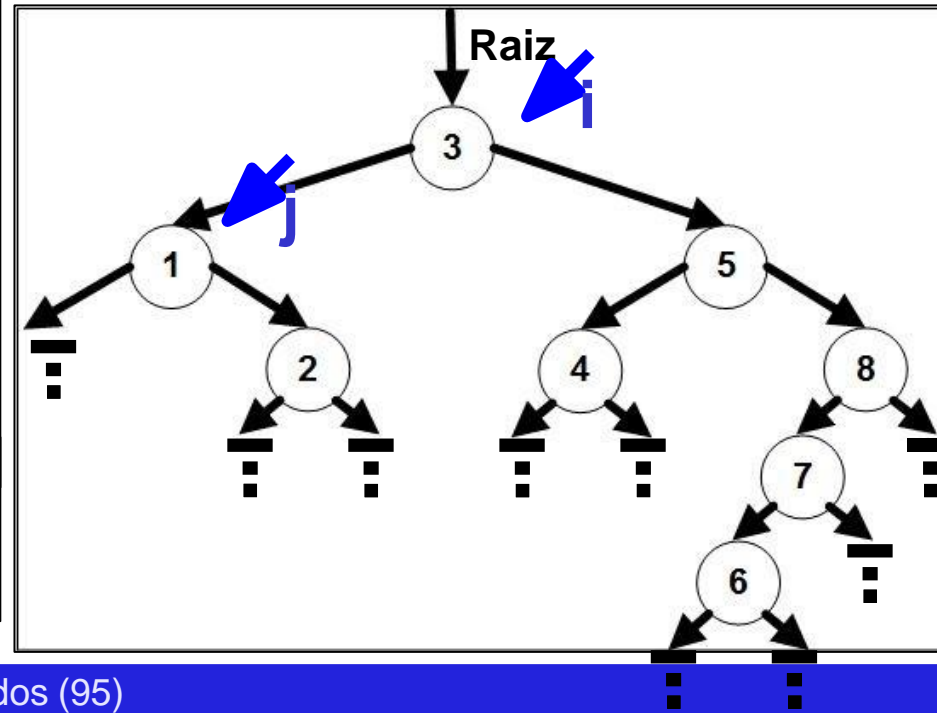
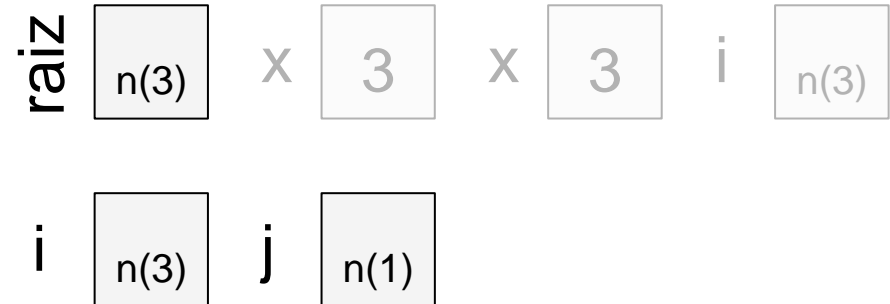
# Algoritmo de Remoção

//Remover(3), dois filhos

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}

No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```



# Algoritmo de Remoção

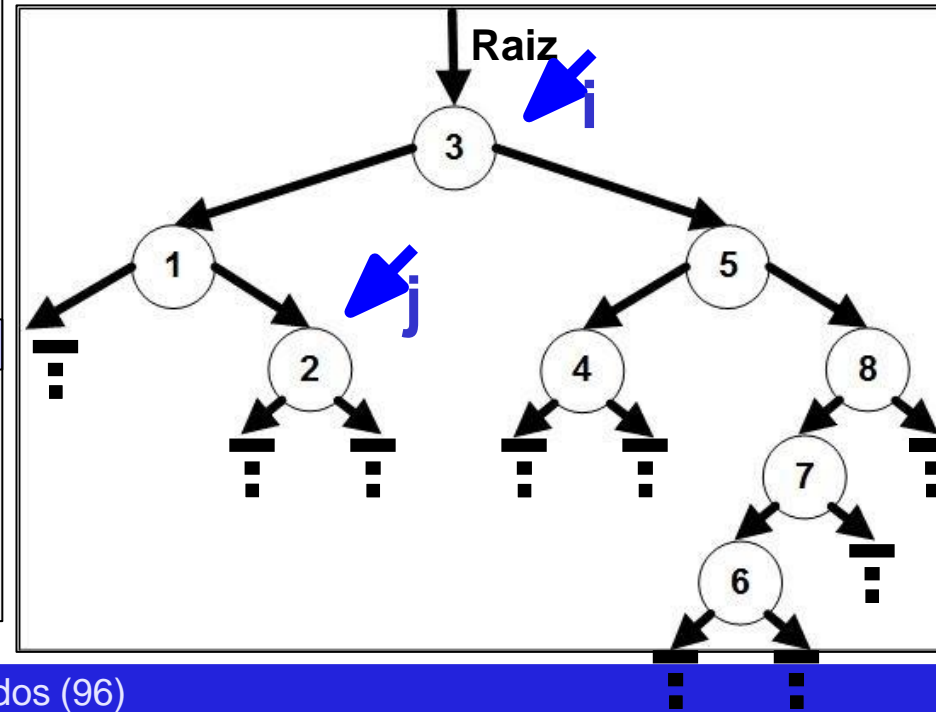
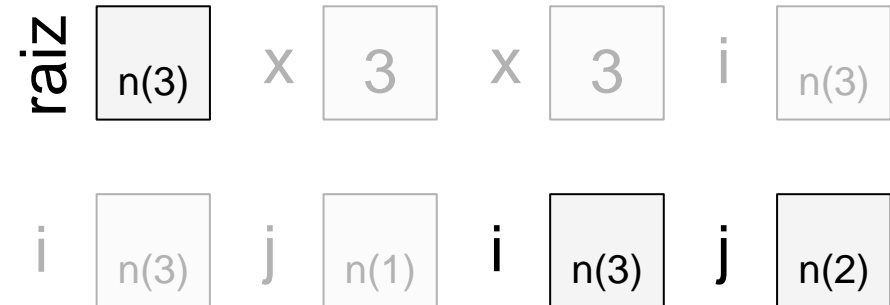
//Remover(3), dois filhos

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}
```

No MaiorEsq(No i, No j) {

```
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```





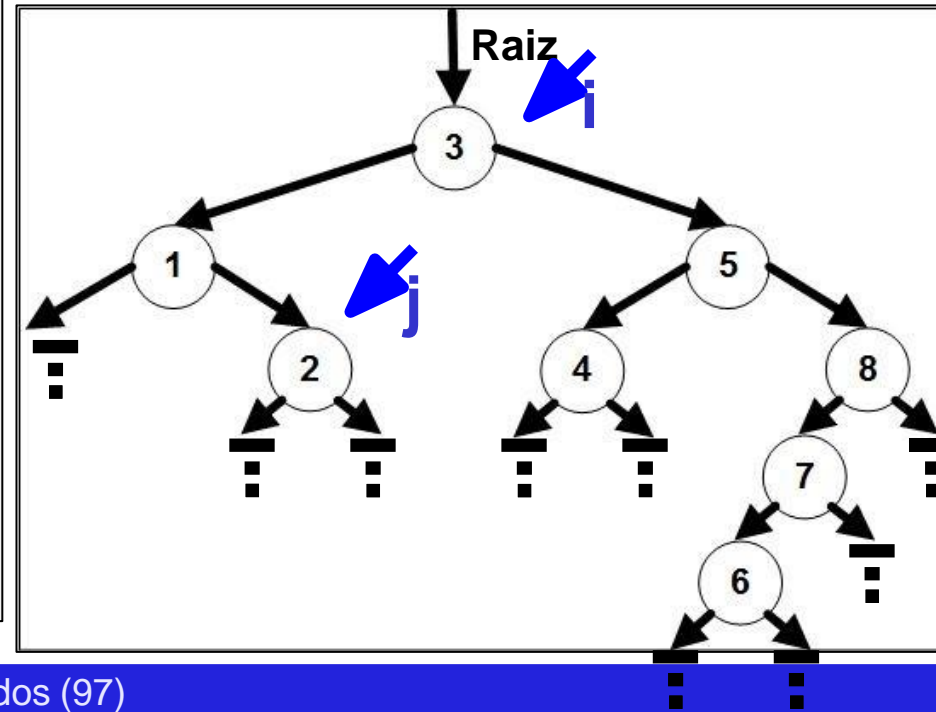
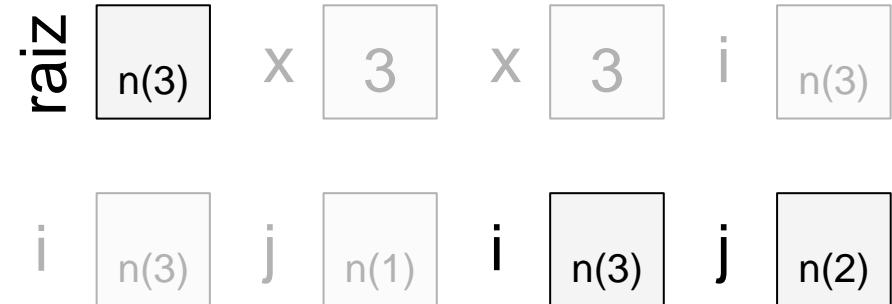
# Algoritmo de Remoção

//Remover(3), dois filhos

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}

No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```



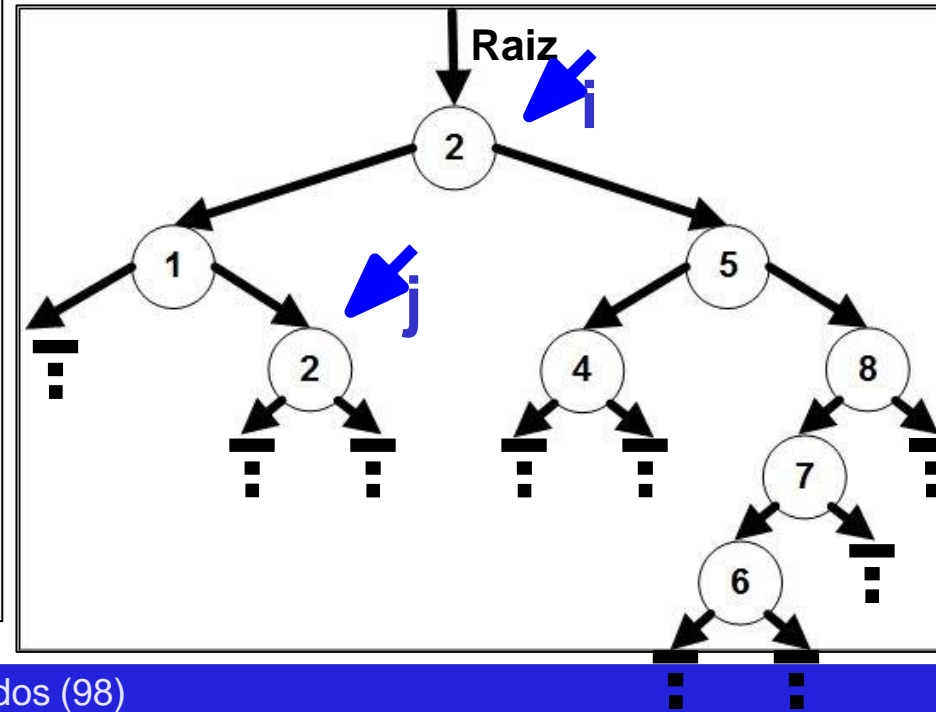
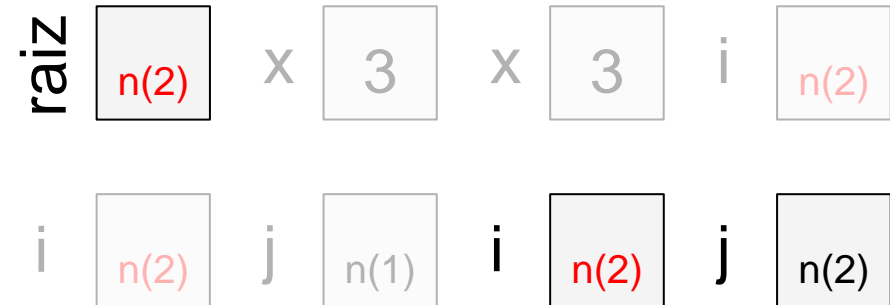
# Algoritmo de Remoção

//Remover(3), dois filhos

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}

No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```



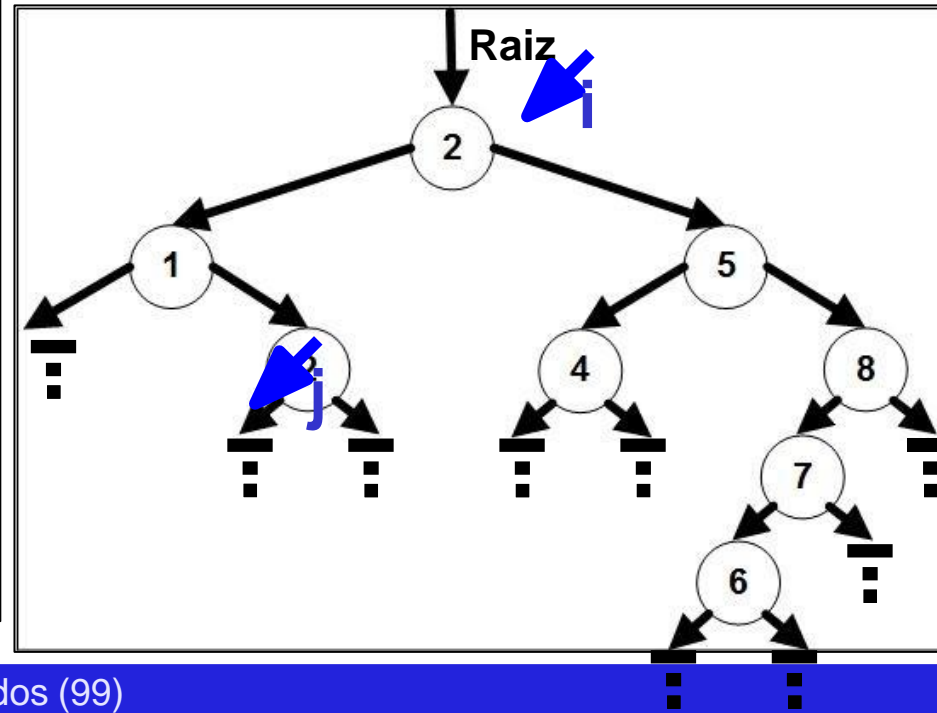
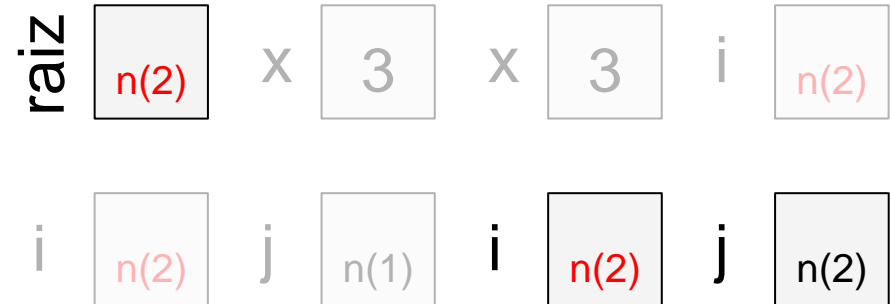
# Algoritmo de Remoção

//Remover(3), dois filhos

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}

No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```



# Algoritmo de Remoção

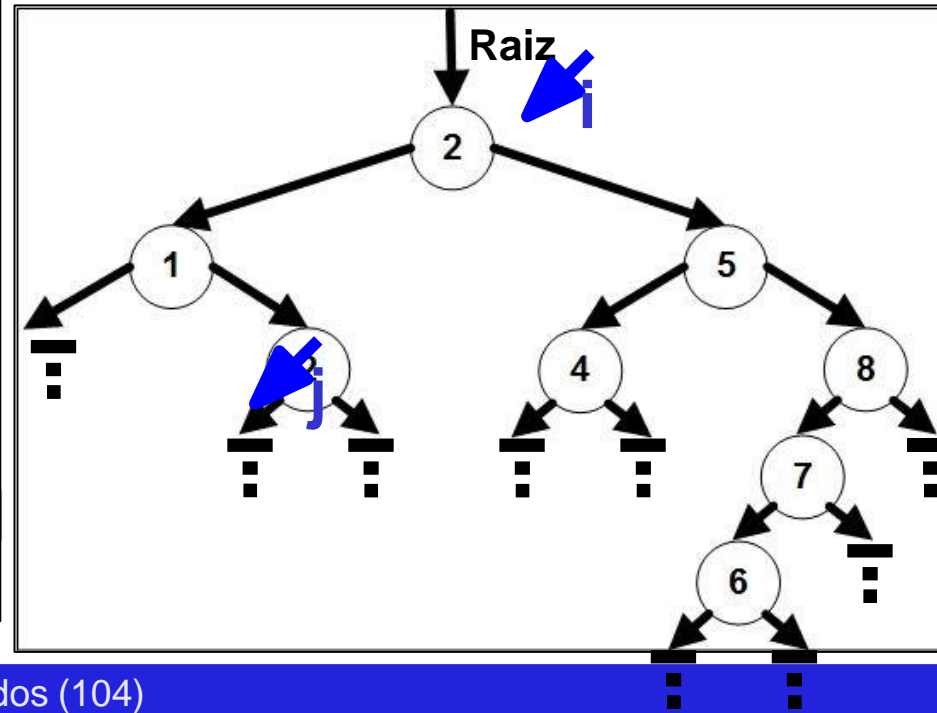
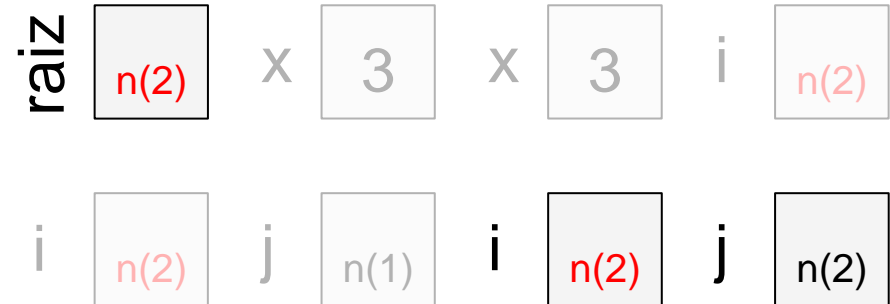
//Remover(3), dois filhos

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}

No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```

Retornando null



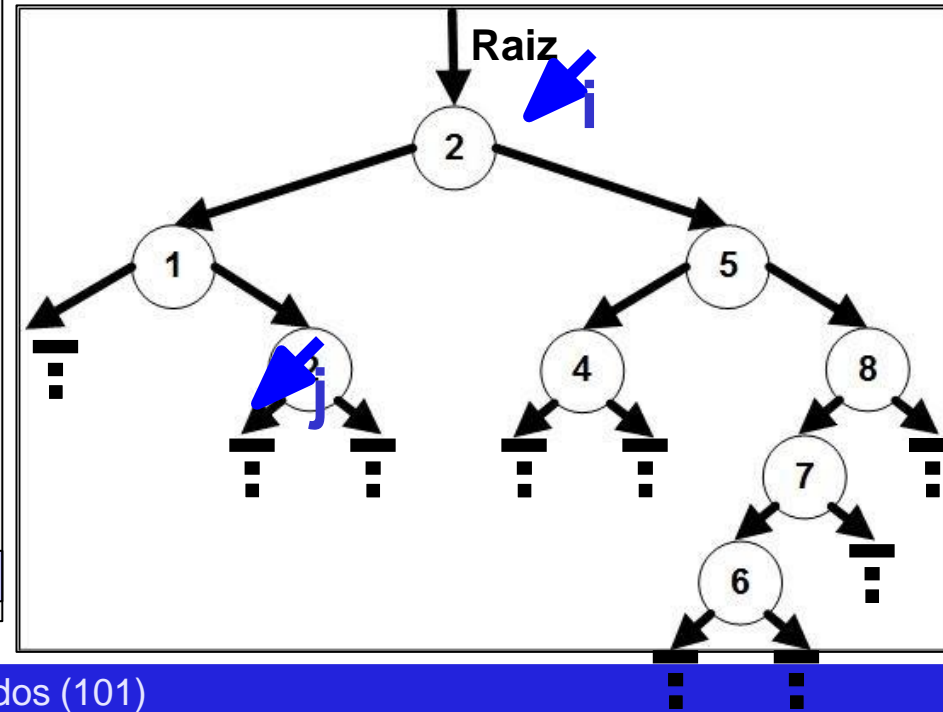
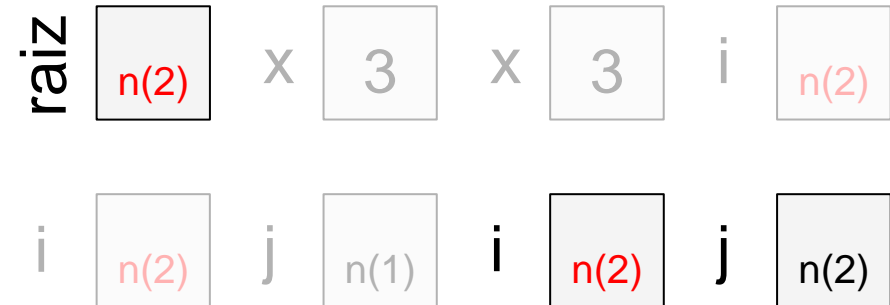
# Algoritmo de Remoção

//Remover(3), dois filhos

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}

No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```



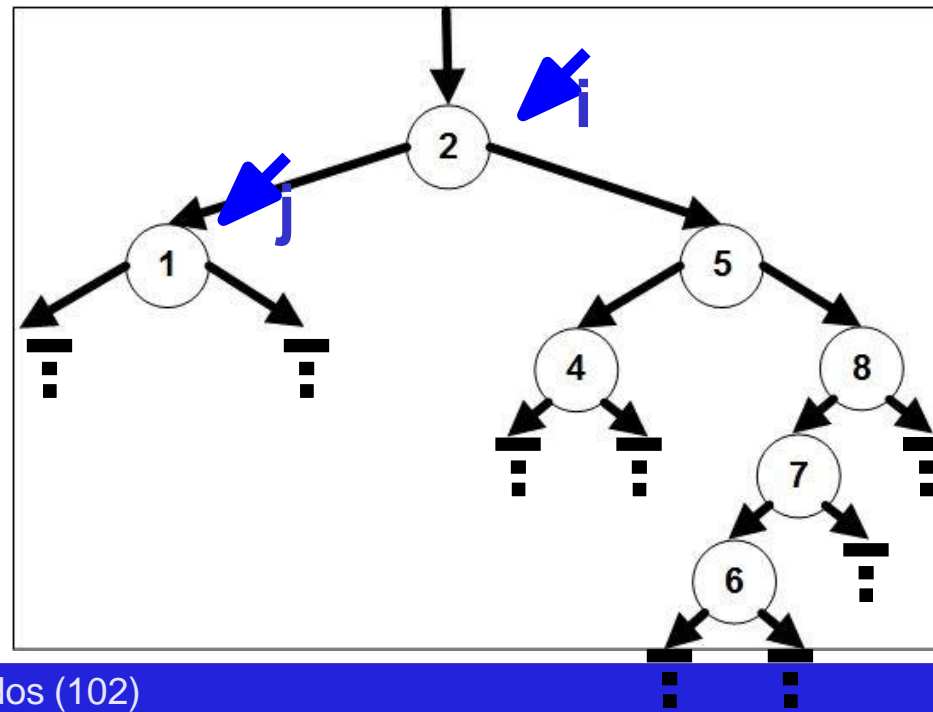
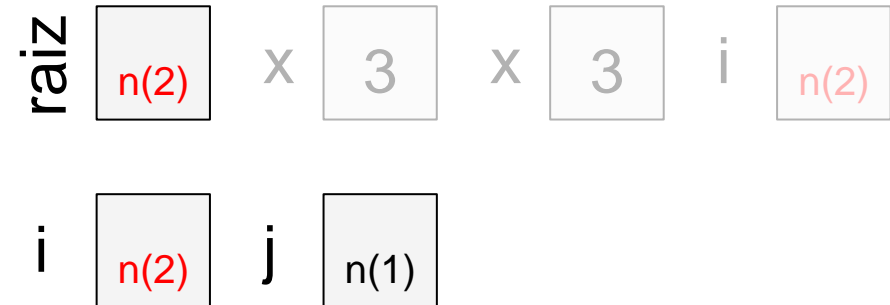
# Algoritmo de Remoção

//Remover(3), dois filhos

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}

No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```



# Algoritmo de Remoção

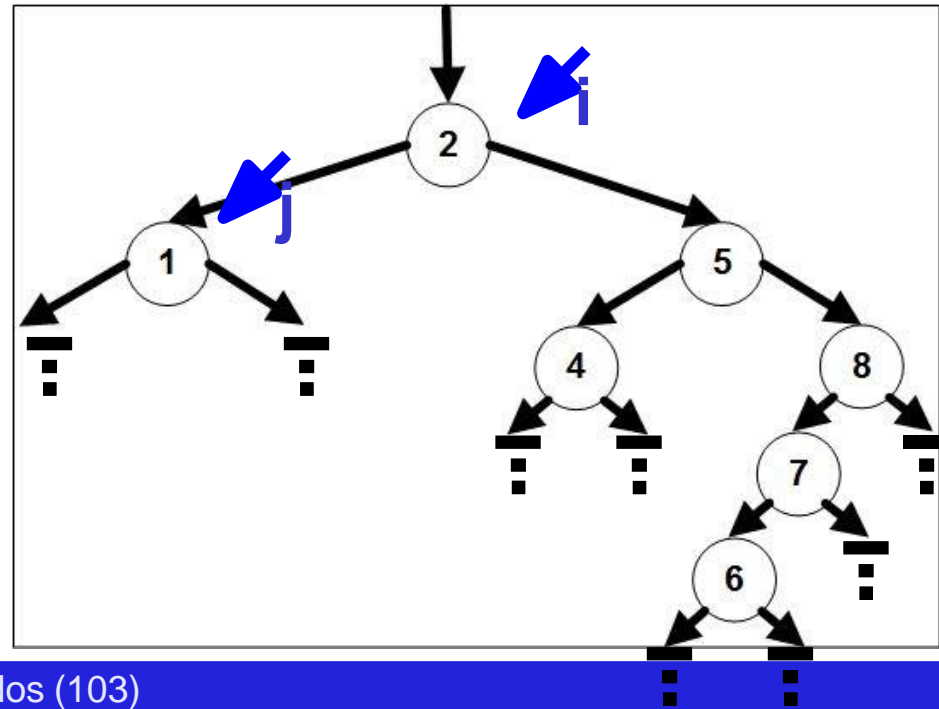
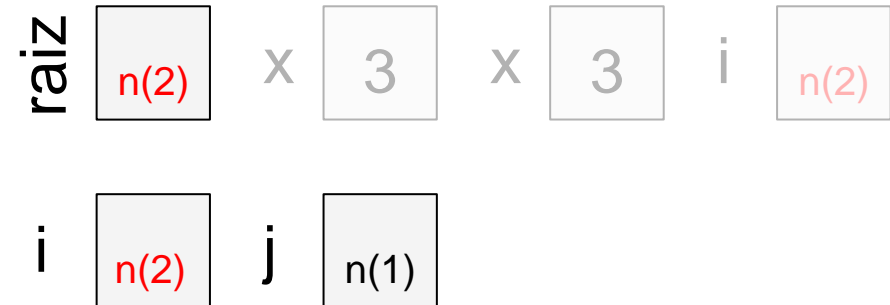
//Remover(3), dois filhos

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}

No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```

Retornando n(1)



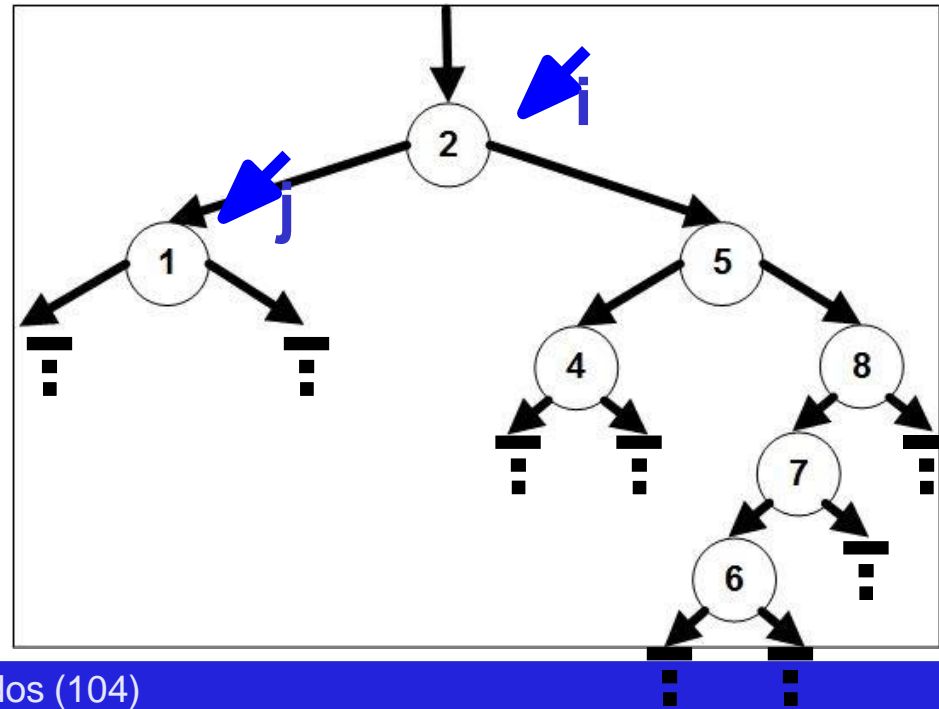
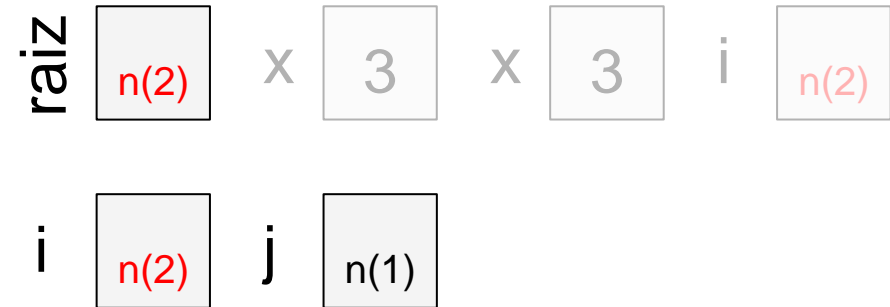
# Algoritmo de Remoção

//Remover(3), dois filhos

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}

No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```





# Algoritmo de Remoção

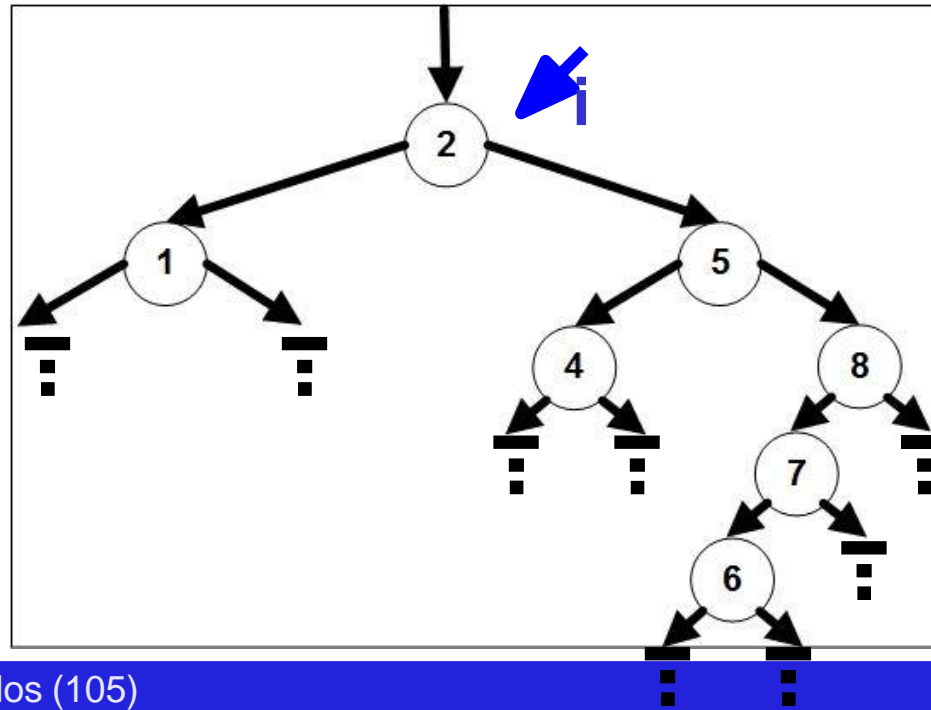
//Remover(3), dois filhos

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}

No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```

raiz n(2) x 3 x 3 i n(2)



# Algoritmo de Remoção

//Remover(3), dois filhos

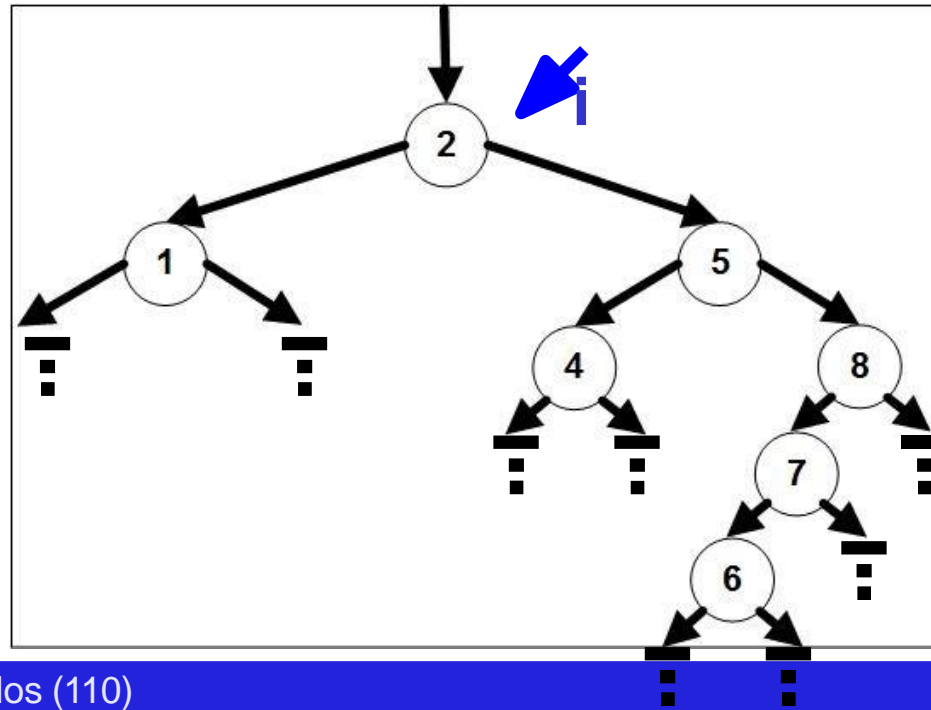
```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}
```

Retorna n(2)

```
No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```

raiz n(2) x 3 x 3 i n(2)



# Algoritmo de Remoção

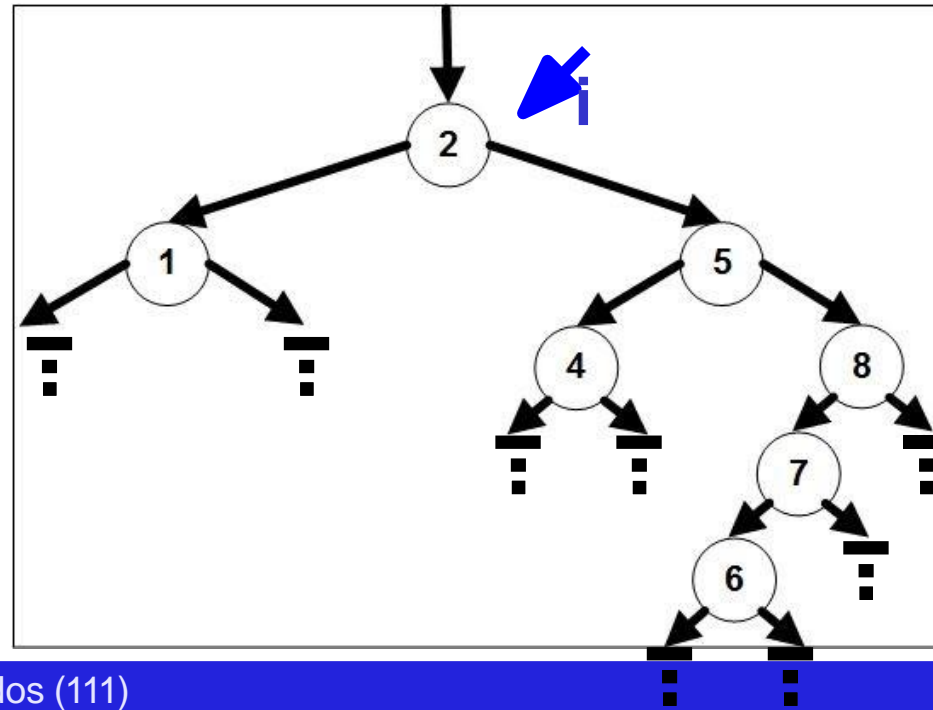
//Remover(3), dois filhos

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}
```

```
No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```

raiz n(2) x 3 x 3 i n(2)



# Algoritmo de Remoção

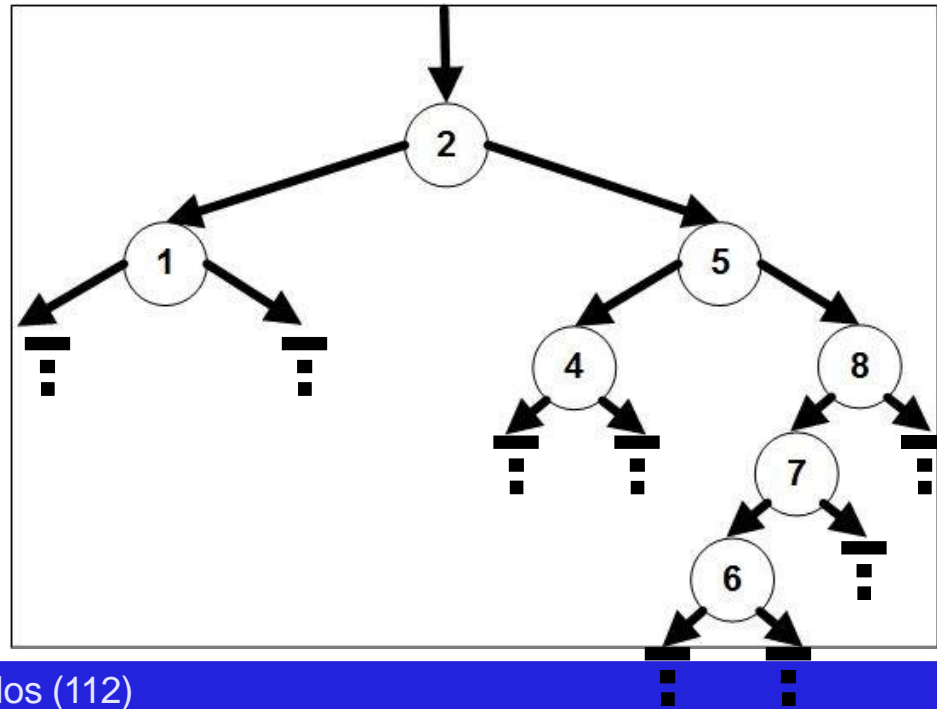
//Remover(3), dois filhos

```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}

private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}

No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```

raiz n(2) x 3



# Algoritmo de Remoção

//Remover(3), dois filhos

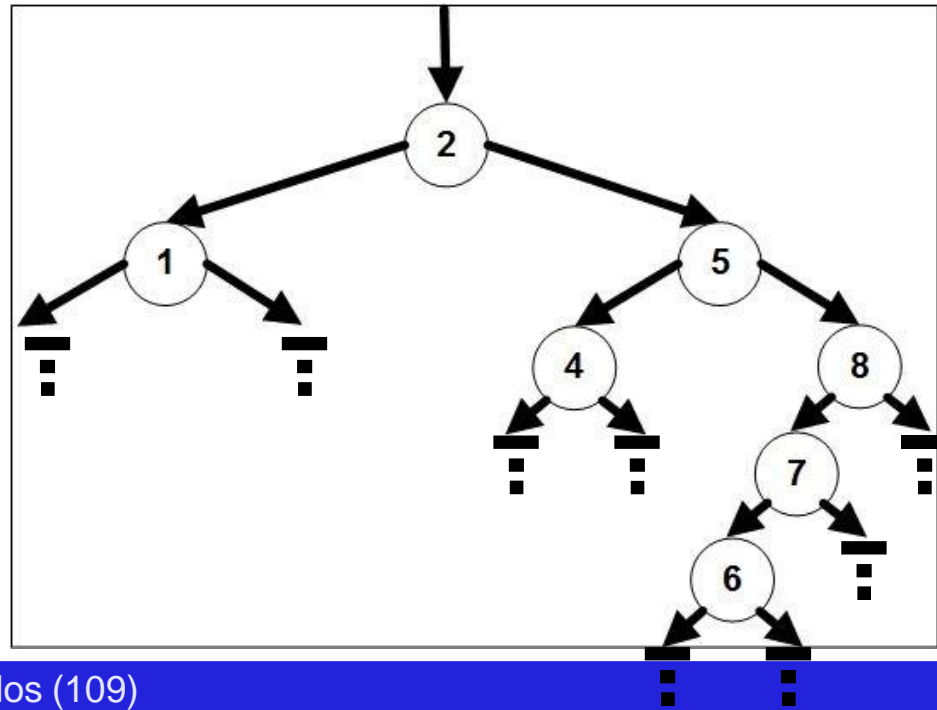
```
public void Remover(int x) {
    raiz = Remover(x, raiz);
}
```


```
private No Remover(int x, No i) {
    if (i == null) { throw new Exception("Erro!"); }
    else if (x < i.Elemento) { i.Esq = Remover(x, i.Esq); }
    else if (x > i.Elemento) { i.Dir = Remover(x, i.Dir); }
    else if (i.Dir == null) { i = i.Esq; }
    else if (i.Esq == null) { i = i.Dir; }
    else { i.Esq = MaiorEsq(i, i.Esq); }
    return i;
}
```

```
No MaiorEsq(No i, No j) {
    if (j.Dir == null) { i.Elemento = j.Elemento; j = j.Esq; }
    else { j.Dir = MaiorEsq(i, j.Dir); }
    return j;
}
```

raiz

n(2)



- Funcionamento básico
- Algoritmo
- **Análise de complexidade** 

# Análise de complexidade da Remoção

- **Melhor Caso:**  $\Theta(1)$  comparações e acontece, por exemplo, na raiz
- **Pior Caso:**  $\Theta(n)$  comparações e acontece, por exemplo, quando inserimos os elementos em ordem e o elemento a remover está na folha
- **Caso Médio:**  $\Theta(\lg(n))$  comparações e acontece, por exemplo, quando a árvore está balanceada e desejamos remover um elemento localizado em uma das folhas